Developer Driven and User Driven Usability Evaluations

Bruun, Anders

Publication date:
2013

Document Version
Publisher's PDF, also known as Version of record

Developer Driven and User Driven Usability Evaluations

By Anders Rysholt Bruun

Department of Computer Science

Aalborg University

Foreword

A special thanks goes to my supervisor, professor Jan Stage, who, like myself, enjoys the spirit and special treats of Western Jutland:

Det var en søndag nat
Jo - der var ingen pjat
Vi var ue mæ æ snørre
De hang å æ snor
De var særlig goe i or
Men vi stjal kun dem som var tørre

De tørre er de best
Men de våde er de flest
Vi plejer no å æde em allivel
Nåve hår fåt for møj
Å anner for lidt
Men de røe er de best uden tvivl

Vi var nok nove grise
Såen som vi ku' spise
Men det er svær å stop
De hang ue a æ mund
Men vi vist det var sund
Så vi åd bare nove flere end vi ku' prop

De tørre er de best
Men de våde er de flest
Vi plejer no å æde em allivel
Nåve hår fåt for møj
Å anner for lidt
Men de røe er de best uden tvivl

- Johnny Madsen, De Tørre Er De Best

English Abstract

Usability evaluation provide software development teams with insights on the degree to which a software application enables a user to achieve his/her goals, how fast these goals can be achieved, how easy it is to learn and how satisfactory it is in use Although usability evaluations are crucial in the process of developing software systems with a high level of usability, its use is still limited in the context of small software development companies .

Several approaches have been proposed to support software development practitioners (SWPs) in conducting usability evaluations, and my thesis explores two of these:

1) The first approach is to support SWPs by training them to drive usability evaluations.

2) The second approach to support SWPs involves minimalist training of end users to drive usability evaluations.

In related work, a set of five quality criteria for usability evaluations is applied to measure performance of usability evaluation efforts. These criteria cover thoroughness, validity, reliability, downstream utility and cost effectiveness.

This leads to my overall research question: *Can we provide support that enables software development practitioners and users to drive usability evaluations, and how do they perform with respect to the quality criteria?*

I studied the developer driven and user driven approaches by firstly conducting literature surveys related to each of these topics followed by artificial settings research and finally by conducting research in natural settings. The four primary findings from my studies are: 1) The developer driven approach reveals a high level of thoroughness and downstream utility. 2) The user driven approach has higher performance regarding validity 3) The level of reliability is comparable between the two approaches. 4) The user driven approach will, arguably, outperform the developer driven in terms of cost effectiveness.

Dansk Resumé

Usablityevaluering giver teamet i et softwareudviklingsprojekt indblik i, hvorledes en applikation lader brugeren opnå sine mål, hvor hurtigt målene opnås, hvor let det er at lære, at bruge applikationen samt hvor tilfredsstillende applikationen er i brug. Når målsætningen er at udvikle en applikation med et højt niveau af usability, så er usabilityevaluering en vigtig aktivitet at gennemføre. På trods af dette er brugen af denne form for evaluering begrænset, specielt indenfor konteksten af små virksomheder.

Litteraturen foreslår flere forskellige tilgange til, hvor udviklingsteams kan understøttes, når de skal udføre usabilityevalueringer. I denne afhandling fokuserer jeg på to af disse:

1) Understøttelse af udviklerne via træning i at drive usabilityevalueringer.

2) Understøttelse af udviklerne ved at at give slutbrugeren minimal træning i at drive usabilityevalueringer.

Litteraturen nævner derudover fem kriterier, som kan benyttes til at vurdere kvaliteten af udførelsen af usabilityevalueringer: Grundighed, korrekthed, pålidelighed, effekt på applikationen og effektivitet.

Dette leder til mit overordnede forskningsspørgsmål: *Kan vi understøtte udviklingsteams og slutbrugere i en grad, der gør dem i stand til drive usabilityevalueringer, og hvor godt udføres evalueringerne mhp. kvalitetskriterierne?*

Forskningsspørgsmålet er besvaret ved først at undersøge eksisterende litteratur indenfor emnet efterfulgt af forskningsstudier foretaget i hhv. kunstige og naturlige omgivelser. De fire primære resultater fra forskningen viser: 1) Den udviklerdrevne tilgang leverer et højt niveau af grundighed og har en høj effekt på udviklede applikationer, 2) den brugerdrevne tilgang viser et højere niveau af korrekthed og 3) de to tilgange viser sammenlignelige niveauer af pålidelighed og 4) den brugerdrevne tilgang har sandsynligvis et højere niveau af effektivitet.

Table of Contents

1 Introduction

Usability is a quality attribute of software applications that determines "the effectiveness, efficiency and satisfaction with which specified users can achieve specified goals in a particular environment" [30]. Usability evaluation provide software development teams with insights on the degree to which a software application enables a user to achieve his/her goals, how fast these goals can be achieved, how easy it is to learn and how satisfactory it is in use [47]. Although usability evaluations are crucial in the process of developing software systems with a high level of usability, its use is still limited in the context of small software development companies [1].

Evaluating the usability of software applications can be accomplished through the use of several methods and methods can be categorized according to their empirical basis. Rubin for instance emphasizes user based evaluations in which users are observed by usability specialists while they use an application to solve a set of predefined tasks and think aloud [47]. Other evaluation methods are based on usability specialists or domain experts inspecting an interface in order to uncover potential usability problems, e.g. Heuristic Evaluation as proposed by Nielsen [42]. There are several approaches to organize the responsibilities of conducting usability evaluations in the context of software development projects. One way is to apply an integrated approach where usability specialists, that are part of the software development team, act as evaluators of their own software [28]. Another is the separate unit approach in which usability specialists from a different organizational unit within the company conduct usability evaluations as a service to the development team [28]. Outsourcing denotes a third approach where usability specialists from another company are hired as external consultants to conduct usability evaluations [28]. The most common way to provide feedback from these approaches is a written report presenting the usability problems experienced by users [29].

At least four approaches have been proposed to support software development practitioners (SWPs) in conducting usability evaluations: 1) The first form of support is to provide SWPs with either software tools or conceptual tools to assist in identifying usability problems. Howarth and colleagues for instance describe the development and evaluation of a software tool with the aim of easing the transformation of raw usability data into usability problem descriptions [25; 26]. Skov and Stage conducted a study in which they developed and evaluated a conceptual tool to support problem identification [51]. 2) The second approach is to provide support to SWPs through usability evaluation methods. As an example, Koustabasis and colleagues conducted an experiment focusing on evaluating the performance of students with respect to number of identified problems, validity and efficiency [33]. An older example of applying the method approach to support SWPs is Nielsens study of the performance of specialists, non-specialists and double-experts in conducting Heuristic Evaluation [42]. 3) The third approach is to support SWPs through training. Häkli presents a study in which she trained software development practitioners without a usability background to conduct Heuristic Inspections and user based tests [21]. Additionally, Høegh and colleagues conducted a study of usability evaluation feedback formats in which they examine how to increase such practitioners' awareness of usability problems [29]. One of these feedback formats was to let the practitioners observe user based evaluations to further involve them in the process [29]. 4) The fourth approach to support SWPs involves minimalist training of end users to conduct usability evaluations. This was proposed by Castillo et al. as a feasible alternative to traditional usability evaluations conducted by usability specialists [10]. One of the main purposes of their User Reported Critical Incident (UCI) method was to reduce the amount of resources required to analyze usability data. Initially users would receive minimalist training in identifying and describing usability problems after which they would report the problems experienced using an online form [10].

There are several causes for the limited application of usability evaluations in small companies. Perceived resource demands and "developer mindset" are two of the primary barriers for conducting usability evaluations [1], [2]. Perceived resource demands is a barrier especially present within small software companies as these do not have the funds to pay for comprehensive consultancy or staffing of usability specialists [21], [31], [50] as they are expensive to hire [44]. The barrier of "developer mindset" reflects the situation that some software development practitioners (e.g. developers) experience difficulties in thinking like a user and what they are capable of [2]. Developer mindset also covers the aspect of acceptance where problems identified through usability evaluations are not always accepted by people in the organization that did not participate in the conduction of the evaluations [2]. Prioritization of fixing identified usability problems is also part of the developer mindset where implementation of functionality and fixing bugs receive higher priority [2]. It can be argued that, if software development practitioners or end users are able to conduct evaluations it could lessen the need for small companies to employ usability specialists. This could potentially solve issues in relation to funding. Also, letting software development practitioners conduct usability evaluations would provide them with first hand observations of users, which in turn could be a solution to overcome the barrier of developer mindset.

1.1 Developer Driven Usability Evaluations

As suggested above, one possible solution to support SWPs could be to train them to conduct evaluations and analyze the data, i.e. to let such practitioners *drive* the evaluations. This is similar to the idea behind the barefoot doctors that emerged during the Cultural Revolution in China in the 1960's, cf. [11]. Getting health care services embedded in the rural areas of China was an ongoing challenge dating back to the early 20th century [11]. Early attempts of solving this challenge included drafting doctors from private practices, but health care services in these areas remained scarce. In 1964 the state covered health care expenditures for 8.3 million urban citizens, which exceeded expenditures of more than 500 million peasants residing in rural areas [11]. Mao Zedong criticized this urban bias of health care services and in 1965 he emphasized the importance of solving this challenge. Thus, one vision behind the Cultural Revolution was to bring better healthcare services to rural areas [11]. To counter this problem, Mao sent mobile teams of doctors into these areas with the purpose of training local peasants in basic medicine such as the delivery of babies, how to ensure better sanitation and how to perform simple surgical procedures [11]. In order to keep up the level of mass production, peasants, who received medical training, would generate work points from their medical services as well as they would receive points for doing agricultural work. Thus, some of the peasants would work part time in the rice fields walking around barefooted and part time as doctors in the local area, which coined the term of barefoot doctors. Although barefoot doctors did not have the same level of competences and equipment as urban doctors, the barefoot programme did, according to the World Health Organization (WHO), effectively reduce health care costs as well as provide timely care [11]. Thus, the WHO considered the barefoot doctors programme successful in terms of solving the challenge of health care shortages [11]. In 1975 Deng Xiaoping suggested that the barefoot doctors gradually should "put on shoes" by improving their medical skills and, hence, move away from agricultural production. This was, however, criticized as this would go against the political goals of mass production [11].

1.2 User Driven Usability Evaluations

An alternative approach is to let the users drive usability evaluations in the sense that they conduct these as well as analyzing the data. The idea behind this approach is not new and in this thesis I use the idea of crowdsourcing as an inspiration. Doan and colleagues describe this approach by applying the notion of systems, e.g. they identify several types of crowdsourcing systems for the web and identify benefits as well

as challenges in relation to these [12]. The overall purpose of crowdsourcing systems is to recruit several users to help solve different types of problems as is the case for e.g. Wikipedia and systems for open source software development [12]. A crowdsourcing system is defined as a system that *"… enlists a crowd of humans to help solve a problem defined by the system owners"* [12]. Various crowdsourcing systems exist and enable different kinds of user contributions. Some crowdsourcing systems enable users to evaluate artifacts such as books or movies and others let users share content in the form of products or knowledge [12]. An example of sharing a product can be found in peer-to-peer services where users share music files and Wikipedia is a classical example of users sharing knowledge. Other crowdsourcing systems enable social networking between users while others depend on users building artifacts, e.g. open source software or execute tasks [12]. Four challenges to crowdsourcing exist. The first challenge relates to how users can be recruited and retained for which there are four solutions: Requiring users to participate, paying users, asking users to volunteer and to make users pay for a service. Given the vast amount of users on the web, the solution of asking for volunteers is mentioned as being free and easily executed, which makes this the most popular approach [12]. The second challenge is related to the types of contributions that users can make. Doan and colleagues mention that, in principle, any non-trivial problem can benefit from crowdsourcing. However, it is important to consider how cognitive demanding the contributions are in relation to the types of users providing these [12]. The third challenge is that of combining contributions, which is a relatively simple task when users provide quantitative data, e.g. numeric ratings, as this can be done automatically. However, when users provide qualitative contributions such as free form texts, a higher degree of manual labor is required [12]. Finally, the fourth challenge is that of evaluating users and their contributions. As crowdsourcing is based on contributions from several users, of which some may be naive, there is a need to filter the data. One solution to this can be to delimit certain types of users of making complex contributions and other solutions include manual or automatic detection of users providing malicious content, e.g. where the system asks questions to which answers are already known [12].

The idea of user driven usability evaluations is similar to the idea of crowdsourcing as a group of users are enlisted in order to solve a problem. The problem to be solved in this case is the identification of usability deficiencies where users are recruited to evaluate an artifact in the form of a user interface. This fits well to the application of the UCI method proposed by Castillo and colleagues [10].

1.3 Quality Criteria for Usability Evaluations

Although usability evaluation methods differ considerably in how they are conducted, Hartson et al. [19] define a set of quality criteria for evaluating the performance of various methods. These criteria are based on the work by Gray and Salzman [17] and Lund [36] who made the point that previous comparison studies of usability evaluation methods (in the late 1990's) had considerable metric related flaws in their experimental designs, i.e. the measured criteria were either "deficient" or "contaminated" to use the words of Hartson et al. [19]. Hartson and colleagues have compiled five quality criteria, which can serve as metrics on how to evaluate and compare the performance of different usability evaluation methods.

Thoroughness

Thoroughness is defined as the proportion of "real" usability problems found using an evaluation method out of the total set of real problems. Determining realness of problems can be accomplished in several ways one of which is by producing a standard list of usability problems. Hartson and colleagues note that conventional laboratory-based usability evaluation based on video analysis can be regarded as the "gold standard" for creating such a list. I define the set of real usability problems as those identified by applying this particular evaluation method. For instance, if usability evaluation method P finds |P| problems and

|LAB| is the set of real problems (identified using conventional laboratory testing), then thoroughness is defined as the set of problems |P| intersecting with |LAB| divided by |LAB|.

Validity

The measure of validity is defined as the proportion of real usability problems found by a particular evaluation method P out of the total identified using this method. In other words, validity is the set of problems |P| intersecting with |LAB| divided by |P|.

Reliability

Reliability is a measure of the consistency in identifying usability problems between multiple evaluators applying a particular evaluation method. In my thesis I apply the any-two agreement as a measure of reliability, cf. [23]. The any-two agreement is an expression of the proportion of problems in common between two evaluators out of the total identified. As an example, if you have two evaluators where the first evaluator uncover the set of problems $|P_1|$ and evaluator 2 uncovers $|P_2|$, then the any-two agreement between these is defined as the intersection of $|P_1|$ and $|P_2|$ divided by the union of $|P_1|$ and $|P_2|$. If there are more than two evaluators, the any-two agreement is defined as the average of the intersection of $|P_i|$ and $|P_j|$ divided by the union of $|P_i|$ and $|P_j|$ for all pairs of evaluators.

Downstream Utility

Downstream utility is a measure of the extent to which results from usability evaluations impacts the usability of a system. Throughout the literature this has been measured in two ways, where the first is the proportion of identified usability problems that development practitioners are committed to fixing. This is denoted the Committed Impact Ratio (CIR). The second way to measure downstream utility is through the proportion of identified usability problems that have actually been fixed, which is denoted the Completed-to-Date Impact Ratio (CDIR), cf. [19, 34, 48].

Cost Effectiveness

Hartson et al. define cost effectiveness as a combination of an evaluation methods ability to detect usability problems and the cost required to do so. In my thesis I apply the measure of total time spent in conducting and analyzing usability data divided by the number of problems identified using a particular method. This gives the average time spent on identifying each problem.

Hartson et al. [19] define these criteria in relation to usability evaluation methods. In my thesis I am interested in evaluating usability evaluation practice. I do, however, also find these criteria relevant for my case. Appendix B provides the list of equations for calculating quality criteria.

1.4 Research Questions

My thesis seeks to study two approaches for supporting software development practitioners in conducting usability evaluations: 1) Developer Driven Usability Evaluations and 2) User Driven Usability Evaluations. This leads to the following overall research question:

Can we provide support that enables software development practitioners and users to drive usability evaluations, and how do they perform with respect to the quality criteria?

The overall research question has been divided into the three detailed research questions presented below.

Research Question 1

- State-of-the-Art: *What is the state-of-the-art in applying developer and user driven approaches for usability evaluation?*

Research Question 2

- Developer Driven: *How do software development practitioners driving usability evaluations perform with respect to the quality criteria?*

Research Question 3

- User Driven: *How do users driving usability evaluations perform with respect to the quality criteria?*

2 Contributions

To answer my research questions I have written five papers, which constitute my research contributions:

[1] Bruun, A. Training Software Developers in Usability Engineering: A Literature Review. In *Proceedings of NordiCHI*. ACM Press, New York, NY, USA (2010).

[2] Bruun, A. and Stage, J. 2011. Training Software Development Practitioners in Usability Evaluations: An Exploratory Study of Cross Pollination. In *Proceedings of the 5th Workshop on Software and Usability Engineering Cross-Pollination: Patterns, Usability and User Experience (PUX 2011)*.

[3] Short version: Bruun, A. and Stage J. Overcoming the Developer Mindset Barrier towards Usability Evaluations. In *Proceedings of the 35th Information Systems Research Seminar in Scandinavia (IRIS 2012)*.
Extended version: Bruun, A. and Stage, J. Training Software Development Practitioners in Usability Testing: An Assessment Acceptance and Prioritization. Accepted for publication in *Proceedings of the 23rd Australian Computer-Human Interaction Conference (OzCHI)*. ACM Press, New York, NY (2012).

[4] Bruun, A., Gull, P., Hofmeister, L. and Stage, J., 2009. Let Your Users Do the Testing: A Comparison of Three Remote Asynchronous Usability Testing Methods. In *Proceedings of the 27th international conference on Human factors in computing systems (CHI)*. ACM Press, New York, NY, USA (2009).

[5] Bruun, A. and Stage, J., 2012. The Effect of Task Assignments and Instruction Types on Remote Asynchronous Usability Testing. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems (CHI)*. ACM Press, New York, NY, USA (2012).

In my research I studied the developer driven and user driven approaches by firstly conducting literature surveys related to each of these topics followed by artificial settings research and finally by conducting research in natural settings. Benbasat distinguish between the notions of artificial, natural and environment-independent settings to categorize research methods [3]. By conducting research in artificial settings the researcher seeks control by creating a specific environment that bounds the experiment [53]. Natural settings research is conducted to study phenomena in real organizations such as actual businesses, government or non-profit organizations [53]. Environment-independent research is based on the assumption that studied phenomena are unaffected by the research setting like, e.g. literature surveys [53]. The research methods applied are elaborated further in the section “Research Methods”.

The 2x2 matrix shown in Table 1 provides an overview of the relations between contributions where the columns represent the studied usability evaluation approaches. These columns are based on the primary actor that drives the evaluations. The rows represent the settings in which the research is conducted.

	Developer Driven	**User Driven**
Artificial Setting	2 1	4 4, 5
Natural Setting	3	5

Table 1: Relations between research contributions.

2.1 Contribution 1

Bruun, A.. Training Software Developers in Usability Engineering: A Literature Review. In *Proceedings of NordiCHI*. ACM Press, New York, NY, USA (2010).

This contribution presents a comprehensive literature survey of research conducted in the area of training novices in usability engineering (UE) methods. Papers were selected as relevant if they described or focused on training of novices in UE methods. In the paper I define "novices" according to Bonnardel et al. [4] and Howarth et al. [26]:

"Novices are persons with less than one year of job experience related to usability engineering and no formal training in Usability Engineering methods"

A preliminary screening was performed using Google Scholar, as this search engine covers scientific papers from a broad set of publishers and proceedings. The search criteria were based on a full-text search in which all the words "training", "developers" and "usability" were required and resulted in 33,800 records (search conducted December 1st 2009). As it would be too tedious a task to read abstracts from all these papers the first 200 abstracts were read of which 23 potentially relevant papers were selected and read in full. Eight papers were selected as relevant and defined the result of the screening process. Papers referenced in the selected 8 papers from the screening were marked as potentially relevant. In addition the 8 relevant papers were looked up on Google Scholar which provides a utility to identify which papers are citing these. All citations were also marked as potentially relevant. Subsequently abstracts from all referenced and cited papers were read and papers fitting the selection criteria were read in full. This process continued until closure was reached after 8 iterations. A total of 4155 abstracts were read and 286 papers were read in full ending up with 129 actually relevant papers (see list of references in Appendix C). Papers were analyzed in terms of research focus, empirical basis, types of training participants and training costs.

Findings show that the majority of papers focus on development, description and evaluation of university curricula and pedagogical approaches (54 %) and UE methods in isolation from organizational context (36 %). Few of the identified papers (10 %) consider the organizational context and that only 3 of these (2 %) are empirical studies on the subject. The literature survey also showed that 10 % of all papers employ industry practitioners as participants in experiments while the majority of papers apply university students as the empirical basis. Thus, the survey also revealed a need to conduct more studies using industry practitioners as the empirical basis. In the survey I also uncovered the amount of related work on training novices in user based usability evaluation methods. One reason for this is that such methods seem to provide wake-up calls for software companies to start improving the usability of their systems as the direct observations of users raise developer awareness, see e.g. [29] and [49]. The literature survey showed that a single empirical based paper is focusing on training novice practitioners in such a method, which also leaves room for future research in this area.

2.2 Contribution 2

Bruun, A. and Stage, J. Training Software Development Practitioners in Usability Evaluations: An Exploratory Study of Cross Pollination. In *Proceedings of the 5th Workshop on Software and Usability Engineering Cross-Pollination: Patterns, Usability and User Experience (PUX)* (2011).

This contribution extends previous research by applying software development practitioners (SWPs) from industry as the empirical basis, which was noted as a future research need in contribution 1. Prior to the experiment a two-day training course was conducted. This lasted 14 hours and the topic was on how to do user based usability evaluations. Five practitioners from a small software development company (20 employees) participated in the experiment. Two of these had no previous knowledge of usability evaluation and one had theoretical knowledge from her education several years ago. The final two practitioners had theoretical and some practical experiences in conducting usability evaluations. Since most of the practitioners had limited or no prior experience in this area, it was decided to train them in classical video analysis as this enables evaluators to review data repeatedly. The emphasis of this study is based on the usability evaluation conducted by the five practitioners after completing the training course. The evaluation was conducted in the usability laboratory at the university and the system evaluated was a web application partly developed by the company in which the practitioners were employed. The five practitioners had, however, not participated in developing the system. All practitioners took part in planning the test while three took turns in acting as test monitor during the test sessions. The five practitioners and a usability specialist analyzed the obtained video material and described and categorized the usability problems according to the categories of critical, serious and cosmetic, cf. [41]. Three unbiased external raters were asked to evaluate the quality of the problem lists created by the practitioners and a usability specialist. To measure the quality of the lists, raters provided a rating based on a Likert scale ranging from 1 = “Not fulfilled” to 5 = “Fulfilled”. These ratings were given on the attributes of clarity, impact, data support, problem cause and user actions, cf. [9].

Findings show that each practitioner after 14 hours of training were able to identify a mean of 24.2 (48.4 %, SD=8.1) usability problems of which 6.8 (57 %, SD=2.6) were critical, 10 (53 %, SD=3.9) serious and 7.4 (39 %, SD=3.2) cosmetic. In comparison the usability specialist found a total of 31 (62 %) where 6 (50 %) were critical, 12 (63 %) serious and 13 (68 %) cosmetic. A total of 50 usability problems were identified. Using equation (1) from Appendix B the mean thoroughness of the practitioners is calculated as follows:

(1) $$\frac{|P_{Mean\ 1\ SWP} \cap LAB|}{|LAB|} = \left(\frac{24.2}{50}\right) = 0.48$$

The reliability of this approach can be calculated using equation (3), which denotes the mean any-two agreement between the practitioners:

(3) $$Avg.of\ \frac{|P_i \cap P_j|}{|P_i \cup P_j|}\ over\ all\ \frac{1}{2} \cdot n \cdot (n-1)\ evaluator\ pairs = 0.38$$

In practice it can be too resource demanding to utilize five evaluators in analysis of usability data, which is why the thoroughness of each pair of practitioners was taken into consideration. Results show that a pair of practitioners on average identify 35.7 (71.4 %, SD=5.2) problems of which 9.2 (77 %, SD=1.9) were critical, 14.8 (78 %, SD=2.4) serious and 11.7 (62%, SD=3.1) cosmetic. The mean thoroughness for all pairs of practitioners is:

(1) $$\frac{|P_{Mean\ Pair\ SWP} \cap LAB|}{|LAB|} = \left(\frac{35.7}{50}\right) = 0.71$$

This shows that a pair of practitioners can outperform a specialist, which applies to the critical and serious problems while the specialist found more cosmetic problems. However, findings also show that practitioners' problem descriptions were of a lower quality compared to that of the usability specialist with the exception of the two most experienced practitioners. The external raters gave an overall median rating of 2 to the descriptions made by the practitioners whereas the specialist got a 4. Additionally it was found that practitioners were better at providing clear and precise problem descriptions than they were at describing the impact, cause, user actions and providing data support for observations.

2.3 Contribution 3

Short version: Bruun, A. and Stage J. Overcoming the Developer Mindset Barrier towards Usability Evaluations. In *Proceedings of the 35th Information Systems Research Seminar in Scandinavia (IRIS 2012)*.

Extended version: Bruun, A. and Stage, J. Training Software Development Practitioners in Usability Testing: An Assessment Acceptance and Prioritization. Accepted for publication in *Proceedings of the 23rd Australian Computer-Human Interaction Conference (OzCHI)*. ACM Press, New York, NY (2012).

This contribution examines the level of commitment devoted to fix identified usability problems, which is crucial for practice as it is one of the main parts in determining whether a usability evaluation has been a success or not [52]. Previous studies have adopted the concept of "downstream utility" to determine the extent to which results from usability evaluations impacts the usability of a system [19, 34, 48]. Thus, in this study I applied measurements of downstream utility to evaluate the effect of changes made to the system.

The overall idea was to let three practitioners from the partnering company evaluate two versions of the same system. The three practitioners had previously taken part in the experiment presented in contribution 2. They were asked to evaluate the first version of the system after which they spent 3 months fixing the usability problems identified. The time span of 3 months was selected so that the practitioners had sufficient time to fix the problems. After 3 months a second evaluation was conducted to determine the effectiveness of the fixes. Three usability specialists analyzed the video data obtained from the evaluations to provide a benchmark. Two of these were external and had not otherwise taken part in the experiment. Interviews with the practitioners were conducted to provide a deeper understanding of the factors influencing their prioritization of fixing problems and reasons why some of the same problems recurred in the second version of the system. The system evaluated was developed by the partnering company for which the practitioners in this study were responsible. It was a web application used by administrative staff to register and apply for wage subsidies. Each of the two evaluations was conducted in an office at the company and they were based on Instant Data Analysis (IDA), cf. [32]. A study by Kjeldskov and colleagues has shown that IDA requires considerably fewer resources in terms of analyzing usability data [32], which makes this relevant in the context of the small partnering company.

Findings show that the practitioners through the use of IDA were able to identify 33 problems in the first evaluation and 35 in the second while the specialists found 31 and 32 respectively. Both groups agreed on 23 problems in each of the two evaluations.

Applying equation (1), this reveals the following thoroughness over the two evaluations:

(1) $$\frac{|P_{Eval_1} \cap LAB|}{|LAB|} = \left(\frac{23}{31}\right) = 0.74$$

(1) $$\frac{|P_{Eval_2} \cap LAB|}{|LAB|} = \left(\frac{23}{32}\right) = 0.72$$

This gives an average thoroughness of 73 % over the two evaluations. Considering the union of problem sets between practitioners and specialists, the practitioners had a thoroughness of 80.5 % across the two evaluations while the usability specialists uncovered an average of 74.5 %. In this case the practitioners found more critical, serious and cosmetic problems than the three specialists, which can be explained by the higher level of domain knowledge of the practitioners compared to the specialists. Additionally, the level of validity can be calculated using equation (2):

(2) $$\frac{|P_{Eval_1} \cap LAB|}{|P_{Eval_1}|} = \left(\frac{23}{33}\right) = 0.69$$

(2) $$\frac{|P_{Eval_2} \cap LAB|}{|P_{Eval_2}|} = \left(\frac{23}{35}\right) = 0.66$$

In terms of downstream utility it was found that the practitioners committed to fixing 20 of the 41 identified problems found in the initial version of the system, which according to equation (4) gives the following Committed Impact Ratio (CIR):

(4) $$CIR = \frac{No.of\ problems\ comitted\ to\ fix}{No.of\ identified\ problems} = \frac{20}{41} = 0.61$$

An interview revealed that the practitioners mainly committed to fixing problems based on the factors of resource requirements and coherence to other systems. Severity ratings and length of problem descriptions were less influential on their commitment and it did not matter whether a problem was experienced by a single or multiple test users. Findings also show that the practitioners managed to eliminate 21 of the 41 problems found in the first evaluation, which gives the following Completed-to-Date Impact Ratio (CDIR):

(5) $$CDIR = \frac{No.of\ problems\ fixed}{No.of\ identified\ problems} = \frac{21}{41} = 0.64$$

This resembles the downstream utility found in another organizational setting where usability practices already have been established, cf. [22]. During the interview it was revealed that the practitioners had tried to fix most of the problems that recurred, but that these fixes did not work as intended. Additionally, one of the problems was not accepted after occurring in the first evaluation, but was then prioritized after its presence in the second. Although the practitioners managed to eliminate 64 % of the problems found in the initial version of the system, 44 problems were found in the second version. This is similar to the 41 problems found in the first version, which means that a considerable amount of new problems were introduced in the new design. Finally, a plausible downside to letting the software development practitioners evaluate their own systems was discovered. There were considerable disagreements on severity ratings given by the practitioners and usability specialists. In this case practitioners consistently gave lower ratings than the specialists in 46 % of the problems, which could indicate a lack of objectivity within the group of practitioners.

2.4 Contribution 4

Bruun, A., Gull, P., Hofmeister, L. and Stage, J. Let Your Users Do the Testing: A Comparison of Three Remote Asynchronous Usability Testing Methods. In *Proceedings of CHI*. ACM Press, New York, NY, USA (2009).

This contribution presents a literature survey and an empirical study concerning remote asynchronous usability evaluations. The literature survey revealed a total of 22 papers presenting empirically based studies of remote asynchronous methods. The majority of papers (17) present comparative studies of remote asynchronous usability evaluation and more established methods such as conventional user-based evaluations, inspection methods or both. The remaining 5 papers document empirical studies in which an asynchronous method was applied without comparing it to other methods. Five categories of remote asynchronous methods were also identified through the literature survey: Auto logging, User Reported Critical Incident (UCI), unstructured problem reporting, forum and diary. Auto logging is a method where quantitative data like visited URL history and the time used to complete tasks are collected in log files that are analyzed. UCI is based on the idea that the users themselves report the problems they experience, which relieves evaluators from conducting the evaluation and analyzing results. Unstructured problem reporting is based on the idea that participants make notes on the usability problems they encounter while working on a set of tasks. An online discussion forum has also been proposed as a source for collecting usability feedback. Finally, a study applied a diary on a longitudinal basis where participants provided qualitative information.

Based on findings in the literature survey three remote asynchronous usability evaluation methods were systematically compared. The three methods were: User-reported Critical Incident (UCI), forum-based online reporting and discussion, and diary-based longitudinal user reporting. A traditional laboratory-based think aloud method was included as a benchmark. The empirical basis consisted of 40 participants, 10 in each condition, who were all university students. They were asked to evaluate the open source email client Mozilla Thunderbird by solving 9 tasks. The 30 participants applying the remote asynchronous methods were trained in identification and categorization of usability problems via written instructions based on descriptions and examples of what a usability problem is and how it is identified. In all remote asynchronous conditions, participants worked at home using their own computer and they were told that they could carry out the tasks whenever they wanted. Findings show that the laboratory evaluation identified a total of 46 usability problems of which 20 were critical, 14 serious and 12 cosmetic. Participants applying the remote asynchronous UCI method identified a total of 13 problems where 10 were critical, 2 serious and 1 cosmetic. In this case the remote participants and specialists applying the conventional laboratory method agreed on 11 problems, which gives the following thoroughness:

(1) $$\frac{|P_{UCI} \cap LAB|}{|LAB|} = \left(\frac{11}{46}\right) = 0.24$$

In the forum-based condition a total of 15 problems were uncovered of which 9 were critical, 1 serious and 5 cosmetic. There was an agreement on 13 problems between the laboratory and forum-based conditions, hereby giving the following thoroughness:

(1) $$\frac{|P_{Forum} \cap LAB|}{|LAB|} = \left(\frac{13}{46}\right) = 0.28$$

The final remote asynchronous method was the longitudinal diary where participants found a total of 29 problems where 11 were critical, 6 serious and 12 cosmetic. There was an agreement on 15 problems between the laboratory and diary conditions:

(1) $$\frac{|P_{Diary} \cap LAB|}{|LAB|} = \left(\frac{15}{46}\right) = 0.33$$

From equation (2) the validity of the remote asynchronous methods are the following:

(2) $$\frac{|P_{UCI} \cap LAB|}{|P_{UCI}|} = \left(\frac{11}{13}\right) = 0.85$$

(2) $$\frac{|P_{Forum} \cap LAB|}{|P_{Forum}|} = \left(\frac{13}{15}\right) = 0.87$$

(2) $$\frac{|P_{Diary} \cap LAB|}{|P_{Diary}|} = \left(\frac{15}{29}\right) = 0.52$$

Looking at the total time spent on preparation and analysis it was found that the laboratory evaluation required 61 person hours while the UCI method required little more than 8 person hours, the forum required just over 9 person hours while the diary condition required slightly below 19 person hours. The cost effectiveness of each of the methods is derived using equation (6):

(6) $$\frac{Total\ time\ spent_{UCI}}{No.of\ problems\ identified_{UCI}} = \frac{493\ min}{13\ problems} = 38\ min.per\ problem$$

(6) $$\frac{Total\ time\ spent_{Forum}}{No.of\ problems\ identified_{Forum}} = \frac{558\ min}{15\ problems} = 37\ min.per\ problem$$

(6) $$\frac{Total\ time\ spent_{Diary}}{No.of\ problems\ identified_{Diary}} = \frac{1126\ min}{29\ problems} = 39\ min.per\ problem$$

(6) $$\frac{Total\ time\ spent_{LAB}}{No.of\ problems\ identified_{LAB}} = \frac{3663\ min}{46\ problems} = 80\ min.per\ problem$$

Thus, overall the three remote methods performed significantly below the classical lab test in terms of the number of usability problems identified, especially with respect to the serious and cosmetic problems, which may seem disappointing. However, considering the cost effectiveness it is shown that the remote methods required less than 50 % of the time to uncover each problem compared to the laboratory method while providing around 50 % of the critical problems.

2.5 Contribution 5

Bruun, A. and Stage, J. The Effect of Task Assignments and Instruction Types on Remote Asynchronous Usability Testing. In *Proceedings of CHI*. ACM Press, New York, NY, USA (2012).

This contribution presents a literature survey and a field experiment comparing the effect of different instruction types and task types. The literature survey revealed that previous studies primarily comprises of feasibility studies of the performance of remote asynchronous methods compared to user-based laboratory testing or expert inspections. Only one study had focused on different types of instructions. In that study, however, users were physically present during the training sessions, which defies the purpose of remote evaluation. More research was found in relation to task assignments where 9 out of 12 papers provide tasks for users to solve while 3 papers do not. However, none of them compare conditions with and

without task assignments. In this field experiment the variables of task type and instruction type were controlled. Task types consisted of two factors; predefined tasks and no tasks. The type of instruction had three factors; deductive (definition), inductive (examples) and a combination of deductive and inductive instructions which all provided guidance on how to identify and describe usability problems. As a benchmark a traditional laboratory evaluation was conducted. Fifty-three university students undertaking an ICT related education provided the empirical basis. They were asked to evaluate the usability of the website for the School of Information and Communication Technology (SICT) using the User Reported Critical incident method. The SICT website provides information about study regulations, educations, exams, study board members, contact information, campus maps etc. The majority of functionality on the core website enables students to retrieve information about educations, organization etc. Thus, this website is developed and maintained within the university. Participants were instructed to report any usability problem they found on the website as soon as they discovered these, which was done using the same web-based report form applied in contribution 3. Participants in the task based conditions received a list with task assignments while participants in the non-task based were asked to report problems experienced during their daily usage, i.e. when using the system for their own purposes.

Findings show that participants in the task-based conditions attempted to solve a mean of 3.8 tasks (SD=1.96, n=23) while participants in the non-task based attempted to solve a mean of 1.5 tasks (SD=1.04, n=19), a difference which is significant. Participants in task-based conditions identified a total of 29 problems of which 13 were critical, 10 serious and 6 cosmetic. In comparison the laboratory evaluation revealed a total of 36 problems distributed as 12 critical, 13 serious and 11 cosmetic. The laboratory and task-based conditions had an agreement on 23 problems, which gives the following thoroughness:

(1) $$\frac{|P_{Task-Based} \cap LAB|}{|LAB|} = \left(\frac{23}{36}\right) = 0.64$$

Participants that did not receive predefined tasks reported a total of 13 problems where 4 were critical, 6 serious and 3 cosmetic. There was an agreement on 10 problems revealing the thoroughness below:

(1) $$\frac{|P_{No-tasks} \cap LAB|}{|LAB|} = \left(\frac{10}{36}\right) = 0.28$$

Statistical analysis shows a significant difference between the remote task-based and non-task based conditions in terms of thoroughness. There was no significant difference between the conventional laboratory evaluation and the task based condition.

The validity of the task-based and non-task-based conditions is calculated by applying equation (2):

(2) $$\frac{|P_{Task-Based} \cap LAB|}{|P_{Task-Based}|} = \left(\frac{23}{29}\right) = 0.79$$

(2) $$\frac{|P_{Non-Task-Based} \cap LAB|}{|P_{Non-Task-Based}|} = \left(\frac{10}{13}\right) = 0.77$$

Considering reliability, results indicate that participants in remote non-task based conditions have a significantly lower any-two agreement compared to participants in task based conditions and evaluators in the LAB condition:

(3) Avg. any-two agreement over Task-based conditions = 0.28

(3) Avg. any-two agreement over Non Task-based conditions = 0.04

(3) Avg. any-two agreement for laboratory condition = 0.54

Thus, a lack of predefined task assignments increases variation between the usability problems identified, which demonstrates that authentic system use could be well suited for exploratory tests where specific evaluation goals are missing. However, if specific areas in an interface need to be evaluated, users should be given predefined tasks to keep them within these limits.

Additionally, participants in the remote conditions who were instructed by means of inductive examples identified 29 problems where 13 were critical, 10 serious and 6 cosmetic. Participants receiving a combination of a deductive definition and inductive examples found 25 problems distributed as 10 critical, 9 serious and 6 cosmetic. Those receiving a deductive definition found 17 problems of which 9 were critical and 8 serious. Thus, in terms of instructions participants were most effective in identifying problems when they received inductive instructions only, which in turn reveals a significant difference compared to those receiving a deductive definition. Finally it was found that participants rated the inductive instructions higher (median=5) than the deductive (median=4) on a 5 point Likert scale. This difference was also significant.

3 Research Methods

This chapter presents reflections on the research methods chosen to answer my research questions. Table 2 shows an overview of the research methods applied in the individual contributions.

Research Question	Contributions	Research Method	Research Purpose	Research Setting
1) *What is the state-of-the-art in applying developer and user driven approaches for usability evaluation?*	1, 4, 5	Literature Survey	Understanding	Environment-Independent
2) *How do software development practitioners driving usability evaluations perform with respect to the quality criteria?*	2	Laboratory Experiment	Prediction	Artificial
	3	Action Case	Change / Understanding	Natural
3) *How do users driving usability evaluations perform with respect to the quality criteria?*	4	Field Experiment	Prediction	Artificial / Natural
	5	Field Experiment	Prediction	Natural

Table 2: Applied research methods.

3.1 Environment-Independent Settings

Based on the categories of artificial, natural and environment-independent settings applied by Wynekoop and Conger, literature surveys belongs to the latter as this is similar to what they denote as “normative writings” [53]. According to Galliers [15], literature surveys assist the researcher in basing future research efforts on already existing knowledge.

3.2 Artificial Settings

A classical example of a research method applied in artificial settings is the laboratory experiment. By using this method the researcher has maximum control over subject assignments and independent variables [53]. Laboratory experiments belong to the positivist tradition aiming to improve predictive power through reduction. An advantage of the laboratory experiment is that it enables high control over variables and strict observation, which in turn leads to precise measurements [53]. The high level of replicability is also one the strengths in using this method [53]. A major downside is the neglect of interference from real world events, in which case obtained data are decoupled from what could be observed in natural settings [53]. Generalizing findings is also restricted to the sample of e.g. participants brought into the laboratory. Lack of generalizability can be compensated by conducting multiple replications of the experiment using different samples [53].

3.3 Natural Settings

In my research, I have based my reserach in natural settings on Braa and Vidgens framework [7]. This framework emphasizes in-context research, which includes the methods in the category of natural settings research presented by Wynekoop and Conger [53] with the addition of Action Case. Braa and Vidgen state three basic outcomes of applying different research methods: Change, understanding and prediction. The

dynamic in Braa and Vidgens framework is inspired by McGraths view of research as a “three horned dilemma” in which maximizing one desirable research outcome comes at the cost of two others [37]. This means that the researcher can maximize the outcome of e.g. change which will come at the cost of lowered outcomes of prediction and understanding. Alternatively the researcher can seek to heighten the outcome on two horns while fully disabling outcome on the third [7].

Action Case is an example of a research method with the aim of heightening outcome on two horns. It enables the researcher to make a trade-off between being an observer that gains understanding through interpretations while at the same time being an actor that imposes change in practice [7]. Action case lies between the soft case method in which rich data are collected to gain insights and the action research method that supports change in an organizational setting [7]. Thus, the action case provides an outcome of smaller interventions compared to action research while maintaining the outcome of understanding from the soft case method [7]. An advantage of action case research is the possibility to introduce and evaluate changes in a relatively short time span [6]. This, on the other hand, is also a downside as the researcher may not have iterated over steps of design, action, implementation, monitoring and evaluation as rigorously and multiple times as is required when conducting full scale action research [20, 38]. Thus, it can be argued that action case research efforts run a higher risk of not leading to sustainable change. Furthermore, the emphasis on the horns of change and understanding comes at the cost of predictive power, which limits generalizability of the outcome [7].

Field experiments are characterized by the control of variables while maintaining a certain level of realism in the sense that research is conducted in natural settings [7]. Thus, in field experiments the researcher aims to control a small number of variables where the effects may be studied in detail [7], which is why field experiments enable the outcome of prediction. The compromise between control and naturalness makes it possible to generalize results to realistic situations [7]. The tradeoff is that, as experimental control increase, the naturalness will decrease [53]. An advantage of field experiments over laboratory experiments is the increased realism that enables evaluation in a more natural setting than is the case for laboratory experiments. This, however, comes at a cost of possible contamination in the experiment due to lack of environmental control [53], which in turn reduces replicability [7].

3.4 Research Question 1

In order to create an offset for my research efforts I wanted to investigate related work, which provided me with knowledge of the state-of-the-art and an understanding of future research needs. Literature survey is the main method applied in contribution 1 and in contributions 4 and 5 it has been applied as a supplement to other research methods.

3.5 Research Question 2

In order to answer my second research question I initially conducted a study in an artificial setting (contribution 2) followed by a study conducted in a natural setting (contribution 3).

Contribution 2 presents a study based on artificial settings. The natural settings research methods revolving around change and understanding tend to require more time to conduct compared to those situated at the point of prediction. For this reason the goal of the experiment in contribution 2 was to conduct a pilot test to predict the software development practitioners’ performance in identifying and describing usability problems before turning to more demanding research methods. It was decided to conduct a laboratory experiment based on the fact that the software development practitioners had no previous experiences in conducting usability evaluations and there were no dedicated testing facilities in

the partnering company. Thus, in this case a laboratory experiment in an artificial setting was more feasible to conduct than natural settings research. A limitation in contribution 2 is the low level of generalizability due to the strict setup in the usability lab consisting of expensive equipment, which may not be feasible to install in small software companies. Additionally, the experiment is based on participants employed in the partnering company exclusively, which may reveal a different behavior than employees from other companies due to e.g. company cultural differences. The number of participants was also a limiting factor in terms of generalizability. Thus, the predictive power of this laboratory experiment is reduced to the performance one could expect to witness from the particular participants employed in the partnering company. On the other hand, the goal of this experiment was to evaluate whether or not it would be feasible to conduct more time consuming research in natural settings and, given the positive findings in terms of thoroughness found in this laboratory experiment, it was considered relevant to proceed into evaluating performance in-context.

Contribution 3 represents a study in natural settings as a follow-up to the laboratory experiment presented in contribution 2. The prediction posed by the laboratory experiment made me expect a similar performance in-context. Specifically I wanted to examine if the approach of training software development practitioners to conduct usability evaluations would solve the problem situation in the partnering company, i.e. whether or not it would cause positive changes. This leans well against conducting action research where the outcome is change. However, in order to establish the outcome of sustainable change within the partnering company [20] it would require several iterations of planning, design, action, observation and evaluation [38]. Given the time constraints of my Ph.D., action research was considered to be a risk as I was unsure whether I had the time to complete more than one iteration and, if not, I would furthermore run the risk of not gaining an outcome of sustainable change. To reduce this risk I chose to proceed in applying the action case research method, which according to Braa and Vidgen is appropriate for doing small scale interventions [7]. Although an outcome of change and understanding was obtained in this action case research effort, the outcome may only be limited to the particular project and the three practitioners that are part of it. Thus, it is not possible to determine whether the approach of training software development practitioners provides sustainable change on company level, in which case it would be necessary to conduct further studies in other projects and, optimally, conduct full scale action research with multiple iterations of planning, design, action, observation and evaluation [38]. This would be an important next step to validate the approach.

3.6 Research Question 3

To answer my third research question I followed the same approach as I did answering the second question where I initially conducted artificial settings research (contribution 4) followed by research in natural settings (contribution 5).

Contribution 4 is based on remote asynchronous usability evaluation that involves users directly in reporting usability problems. The aim of this study was to identify the most effective remote asynchronous method in terms of thoroughness and resource requirements, i.e. time spent on analysis. Although laboratory experiments are more feasible to conduct than natural settings research, I still chose to setup a field experiment. Other studies of remote asynchronous have indeed conducted method evaluations using laboratory experiments where participants apply the methods under researcher observation in an artificial setting, see e.g. [10]. This, however, defies the purpose of the remote asynchronous method, which is why I chose to conduct a field experiment. Even though the research method in several ways can be considered a field experiment and, hence, belong to the category of natural settings research, there are two constraints

that lead the experimental design towards an artificial setting. One constraint relates to the set of predefined tasks that participants were asked to solve, which compromises naturalness as participants could be forced into artificial usage situations [5]. The fact that participants evaluated an email client with no direct relation to their organization (the university) also contributed with a movement towards an artificial setting. Thus, this experimental design creates boundaries that do not reflect method usage in a natural setting.

Contribution 5 represents a field experiment going in depth with the remote asynchronous method of UCI by studying the effect of variables relevant in a natural setting. One of these variables is the type of task assignments, which, if predefined, can compromise naturalness and lean the experiment towards an artificial setting. Yet, most studies of remote asynchronous evaluation methods employ predefined tasks, cf. contribution 5. For this reason I was interested in studying the effect of not giving task assignments. Additionally I was interested in making participants evaluate a system developed and maintained within their organization (the university). Thus, this contribution was directed towards a more natural setting compared to contribution 4. A major limitation in this study is the nature of the field experiment as a research method. The reduction caused by controlling the variables brings an outcome of predictive power within the particular setup, which has further relevance for practice. However, the field experiment comes at the cost of change and understanding and a next step would be to evaluate the User Reported Critical Incident method using research methods that promote change and understanding such as action research or the hybrid of action case.

4 Discussion

The aim of my Ph.D. project was to examine whether software development practitioners and users could be supported to drive usability evaluations as well to evaluate these two approaches according to the five quality criteria of thoroughness, validity, reliability, downstream utility and cost effectiveness [19].

4.1 Thoroughness

Thoroughness is defined as the proportion of "real" usability problems found using an evaluation method out of the total set of real problems. My studies show that the software development practitioners were able to identify a considerable amount of usability problems. Findings in contribution 2 show that each practitioner on average had a thoroughness of 48 % of all problems. In comparison the usability specialist had a thoroughness of 62 %. Results from that study also show that a pair of practitioners had an average thoroughness of 71 %, i.e. a pair of practitioners was able to outperform one usability specialist in this respect when conducting traditional video based analysis. Contribution 3 shows that three practitioners conducting Instant Data Analysis (IDA) had a thoroughness of 73 %.

The measure of thoroughness used throughout this thesis is based on the assumption that usability problems derived through a conventional laboratory evaluation based on video analysis provides the set of real problems (see equation (1) in Appendix B). In other words, this usability evaluation method is used as the “gold standard” for providing actual criteria, cf. [19]. This also means that the problems found only by, e.g. the practitioners or users applying IDA or UCI, but not by specialists using the conventional laboratory method are discarded when calculating the thoroughness. If the set of real usability problems is defined by the union of all problems it is found in contribution 3 that the three usability specialists uncovered 74.5 % of all problems while the practitioners applying IDA identified 80.5 %. Thus, in this case the practitioners outperformed the usability specialists doing video based analysis. This may seem surprising given the fact that the practitioners received a total of 30 hours of training, but may be explained through their level of domain knowledge which was higher than that of the specialists who in this case fit the notion of being external consultants. According to Bruce and Morris an inherent problem in applying an outsourcing approach is that external consultants lack domain knowledge such as customer requirements [8]. The importance of domain knowledge is also supported in other studies, e.g. Nielsens study of usability specialists, non-specialists and double experts [42]. Findings from that study show that usability specialists found more problems using heuristic evaluation than non-specialists while the double experts found most problems [42]. Additionally, Følstad and Hornbæk conducted a study in which a group of end users acted as domain experts in the conduction of Coorporative Usability Evaluations [14]. That study shows that evaluation output was enriched by including domain experts in the interpretation phase as they provided additional insights in identified problems and helped in uncovering a considerable amount of new problems [14]. Thus, these studies and indicate that domain knowledge plays a key role in the identification of usability problems. This indicates an advantage of the developer driven approach over separate unit and outsourcing approaches where usability specialists are distant from the team that develops the software.

In comparison, the user driven approach evaluated in contributions 4 and 5 reveal a lower thoroughness than the developer driven approach. Findings from contribution 4 show that a laboratory evaluation conducted by usability specialists reveals significantly more usability problems compared to the user driven approaches. Thoroughness of the UCI method is 24 % while the forum and diary based methods reveal 28 % and 33 % of the problems respectively. In contribution 5 the participants in task-based conditions demonstrated a thoroughness of 64 %, which, compared to the conventional laboratory condition, did not reveal a statistically significant difference. However, participants that did not receive predefined tasks had a

thoroughness of 28 %, which was significantly lower compared to the conventional laboratory and task based conditions. A plausible reason for the observed differences in thoroughness between developer driven and user driven approaches could lie in the fact that the software development practitioners received 30 hours of in-person training compared to the written instructions submitted remotely to participants in the user driven studies.

An interesting difference between the two user driven studies is that the UCI method in contribution 4 performed significantly worse than the lab while there were no significant differences between the laboratory condition and the task-based conditions in contribution 5 (where UCI is applied). One of the differences between these studies is the number of participants where 10 users participated in contribution 4 (task-based UCI) and 23 in the task-based condition in contribution 5. For many years one of the most disputed guidelines related to usability testing in practice is the number of users needed to achieve a satisfactory outcome [35]. Lindgaard and Chattratichart analysed usability reports written by nine professional usability teams that tested the same interface and their findings show no significant correlation between the number of users and the number of severe problems identified. However, there was a significant correlation between the number of tasks and the number of problems identified where higher task coverage causes a higher number of problems to be identified [35]. This indicates that task coverage has more impact on the number of identified usability problems than the number of test users. In contributions 4 and 5, however, the participants received the same number of tasks. Thus, although the effect of number of users in usability evaluations has been challenged, the above findings indicate that the number of users in a user driven approach should not be dismissed, i.e. more users seem to reveal more problems in this case. The study in contribution 4 was based on users evaluating an open source email client while the system in contribution 5 was a university website directly related to the domain of the users who, in both studies, were university students. Thus, domain knowledge in relation to the system evaluated could also be accountable for these differences. This is in line with the previously mentioned study conducted by Følstad and Hornbæk, which shows that evaluation output can be enriched by including domain experts as they uncover a considerable amount of new problems [14].

4.2 Validity

Validity is defined as the proportion of real usability problems found by a particular evaluation method. The measure of validity is, like thoroughness, based on the assumption that usability problems derived through a conventional laboratory evaluation defines the set of real problems. In contribution 3 it was found that the 3 SWPs applying the IDA method revealed a validity of 67 %. In comparison, the validity of the user driven approaches in contributions 4 and 5 were higher spanning from 77 % - 87 % with the exception of the diary-based evaluation, which provided a validity of 52 %. According to equation (2), these measurements express the proportion of problems identified by a given approach, which are also found using the conventional laboratory method. Thus, the above findings indicate that the user driven approaches have a higher proportion of problems in common with the conventional laboratory evaluation than the developer driven approach. This could also explain the observed differences in validity between developer and user driven approaches as the specialists in all studies were employed or had previously been studying at the same university in which the user driven experiments were conducted. For this reason the specialists had considerable knowledge of this domain in which evaluated systems were used compared to the domain knowledge in case of the developer driven approach.

As was also mentioned in the discussion of thoroughness above, the consequence of using conventional lab as the “gold standard” will be discarded problems, which are uniquely identified by the SWPs and users.

Assuming that problems identified by practitioners and users are all real problems will, on the other hand, always reveal a validity of 100 % according to equation (2). Thus, validity and thoroughness measurements rely heavily on the assumptions made about realness, but determining the ultimate set of real problems is not possible [19].

4.3 Reliability

The any-two agreement is an expression of the average proportion of problems in common between all pairs of evaluators. The reliability of the practitioners in the developer driven approach was 38 %, which is similar to the 28 % found in the task-based condition of the user driven approach. The reliability was considerably lower (4 %) in the non-task based condition of the user driven approach. These similarities and differences may be explained by the fact that the users in the Developer Driven approach received predefined tasks during the evaluation driven by the practitioners, which was also the case for the task-based condition of the user driven approach. As discussed in contribution 5, a lack of predefined task assignments increases variation in the set of identified usability problems. This in turn indicates that a user driven, non-task based approach is well suited for exploratory tests where specific evaluation goals are missing. On the other hand, if specified areas of an interface need to be evaluated, users should be given predefined tasks to keep them within these limits.

4.4 Downstream Utility

Downstream utility is a measure of the extent to which results from usability evaluations impacts the usability of a system. In terms of downstream utility, it was found that the practitioners committed to fixing most of the identified problems and that they prioritized these based on the factors of resource requirements and coherence to other systems. Additionally, the practitioners managed to eliminate most of the problems, which resembles the downstream utility found in another organizational setting where usability practices had already been established, cf. [22]. These findings, combined with the fact that the practitioners identified a considerable amount of problems, indicates that the development driven evaluations caused the practitioners to accept results from usability evaluations as well as prioritize fixing problems, which deviates from the typical developer mindset described in the literature, cf. [1], [2]. This finding may be explained by the awareness that follows from the direct observation of users interacting with the software application as this provides first hand insights into the usability problems experienced by the users as noted in [29].

As contributions 4 and 5 show, then the user driven approach enables users to identify usability problems, but the outcome is still only a list of problems on which the software development practitioners need to base their improvements. Thus, in this case the software development practitioners do not observe the evaluations. This corresponds to the approach where software development practitioners receive a written list of usability problems, which is the most widely used feedback format [29]. Due to the fact that software development practitioners do not observe the users during interaction, this awareness would arguably be compromised when applying user driven evaluations and, in turn, lead to a low level of downstream utility. However, further studies are needed to validate this claim.

Finally, although the practitioners in the developer driven approach managed to eliminate most of the problems found in the initial version of the system, it was also found that the second version introduced a considerable amount of new problems. This behavior is recognized by Nielsen who argues that design and evaluation should be conducted over several iterations as a new design may introduce new usability problems [43].The number of new problems could be reduced if practitioners not only received training in evaluation, but also in interaction design. As Wixon points out, then it is equally important to tell the

practitioners what to do and not just what is wrong within an interface [52]. Thus, in the future it would be crucial to provide such practitioners with training in interaction design to further increase the impact of usability evaluations.

4.5 Cost Effectiveness

Cost effectiveness denotes the average time spent on identifying each usability problem. Contribution 4 shows that the user driven approach requires considerably fewer resources compared to traditional laboratory evaluations based on video based analysis. Findings show that the total time required to prepare evaluations and analyze results required 61 person hours in case of the traditional laboratory evaluation while the remote asynchronous UCI method required little more than 8 person hours. In relation to crowdsourcing, Doan and colleagues mention that the challenge of combining user input is a relatively simple task when users provide quantitative data such as numeric ratings, as this can be done automatically. Qualitative input such as free form text requires a higher degree of manual labor [12] which could compromise the aim of lowering the amount of required resources through the user driven approach. The usability problems reported by the users in contributions 4 and 5 are qualitative in nature. However, considering the cost effectiveness it is shown that the remote methods required less than 50 % of the time to uncover each problem compared to the laboratory method while providing around 50 % of the critical problems. This also included time spent on filtering valid and invalid problem descriptions and, thus, includes time spent solving the challenge of evaluating qualitative user input in a crowdsourcing approach [12]. In relation to resource demands, all instructions in contributions 4 and 5 were conveyed online in written form, which shows that larger groups of users can indeed receive minimalist training and successfully identify usability problems using few resources.

Considering the developer driven approach it was found that the practitioners were able to identify a large amount of usability problems after receiving 30 hours of training. This shows that such practitioners may obtain considerable competences in what may seem to be a short time frame. On the other hand it may be difficult to overcome the barrier of high resource demands when each practitioner has to spend 30 hours on training. Thus, to avoid this initial overhead of training, it may be more feasible to e.g. apply an outsourcing approach where an external usability specialist with the right competences conducts the evaluations. In the long run, however, it can be argued that developer driven evaluations would require less resources as the hourly rates of consultants can be higher than that of internal employees. A study by Bruce and Morris supports this by mentioning that in-house designers are less expensive to use compared to out-house designers [8]. An additional consideration is that the practitioners in contributions 2 and 3 have various job responsibilities of e.g. systems developers, test managers and project managers. This means that they have other tasks than just conducting usability evaluations, which means that when they spend time on conducting evaluations they cannot fulfill other responsibilities such as implementation and planning activities. These other tasks must then be completed at a different point in time. This resembles the critique raised against Deng Xiaopings suggestion of letting Chinas barefoot doctors gradually “put on shoes” by improving their medical skills, as this moved their responsibilities further away from that of agricultural production [11].

Table 3 summarizes my findings of the developer and user driven approaches in relation to quality criteria.

	Developer Driven	User Driven
Thoroughness	**Contribution 2:** 1 SWP (LAB) = 48 % 2 SWPs (LAB) = 71 % 1 Specialist (LAB) = 62 % **Contribution 3:** 3 SWPs (IDA) = 73 % 3 Specialists (LAB) = 100 %	**Contribution 4:** 10 Users (UCI) = 24 % 10 Users (Forum) = 28 % 10 Users (Diary) = 33 % 3 Specialists (LAB) = 100 % **Contribution 5:** 23 Users (Task Based) = 64 % 20 Users (No-Tasks) = 28 % 3 Specialists (LAB) = 100 %
Validity	**Contribution 3:** 3 SWPs (IDA) = 67 % 3 Specialists (LAB) = 100 %	**Contribution 4:** 10 Users (UCI) = 85 % 10 Users (Forum) = 87 % 10 Users (Diary) = 52 % 3 specialists (LAB) = 100 % **Contribution 5:** 23 Users (Tasks) = 79 % 20 Users (No-tasks) = 77 % 3 Specialists (LAB) = 100 %
Reliability	**Contribution 2:** 5 SWPs (LAB) = 38 %	**Contribution 5:** 23 Users (Tasks) = 28 % 20 Users (No-tasks) = 4 % 3 Specialists (LAB) = 54 %
Downstream Utility	**Contribution 3:** CIR = 61 % CDIR = 64 %	**N/A**
Cost effectiveness	**N/A**	**Contribution 4:** UCI ≈ 8 hrs. (38 min. per problem) Forum ≈ 9 hrs. (37 min. per problem) Diary ≈ 19 hrs. (39 min. per problem) LAB ≈ 60 hrs. (80 min. per problem)

Table 3: Overview of developer driven and user driven approaches in relation to quality criteria.

5 Conclusions

This chapter presents the conclusion in which I firstly answer each of my detailed research questions followed by answers to the overall research question.

5.1 Research Question 1

My first research question is:

- *What is the state-of-the-art in applying developer and user driven approaches for usability evaluation?*

This has been answered by conducting a literature survey on the developer driven approach, which is documented in contribution 1. Additionally, as a part of contributions 4 and 5 I conducted literature surveys on the user driven approach. Findings in contribution 1 show that the majority of the identified papers describe studies of university curricula and pedagogical approaches while few papers consider training novices in an organizational context, of which only 3 are empirical studies on the subject. Additionally, most of the papers apply university students as the empirical basis while few papers employ industry practitioners as participants in experiments. Finally, the literature survey revealed a single empirically based paper focusing on training novice practitioners to drive usability evaluations.

State-of-the-art in relation to the user driven approach is identified in contributions 4 and 5. In contribution 4 the literature survey revealed that the majority of papers on this topic present comparative studies of remote asynchronous usability evaluation and more established methods such as the conventional laboratory evaluation. None of the identified papers were comparing the performance between several asynchronous methods. This survey also revealed 5 categories of remote asynchronous methods: Auto logging, User Reported Critical Incident (UCI), unstructured problem reporting, forum and diary. The literature survey in contribution 5 uncovers the use of different task assignments and instruction types applied in related work. A single of the identified papers focused on evaluating different types of instructions, but in this case the users were physically present during the training sessions. In relation to task assignments it was found that most of the papers provide predefined tasks for users to solve. However, none of them compare conditions with and without task assignments.

5.2 Research Question 2

My second research question is:

- *How do software development practitioners driving usability evaluations perform with respect to the quality criteria?*

To answer this I initially conducted research in an artificial setting followed by research in natural settings. Findings in contribution 2 showed that the average level of thoroughness for a single practitioner was 48 % while a pair of practitioners on average were able to identify 71 % of all problems. It is shown that a pair of practitioners can outperform a specialist in this respect. Findings in contribution 3 show that the practitioners had a thoroughness of 73 % using IDA and this also revealed a validity of 67 %. Contribution 2 show that the five practitioners doing conventional video based analysis had a reliability level of 38 %. Additionally, the practitioners committed to fixing 61 % of the identified problems based on the factors of resource requirements and coherence to other systems. Severity ratings and length of problem descriptions were less influential in this respect and it did not matter whether a problem was experienced

by a single or multiple test users. Also, the practitioners managed to eliminate 64 % of the problems, which resembles the downstream utility found in organizational settings with established usability practices. During the interview the practitioners stated that they had tried to fix most of the problems that recurred, but that these fixes did not work as intended.

Although it was found that practitioners were able to outperform usability specialists with respect to thoroughness, findings in contribution 2 also show that practitioners' problem descriptions were of a lower quality compared to that of a usability specialist. Additionally it was found that practitioners were better at providing clear and precise problem descriptions than they were at describing the impact, cause, user actions and providing data support for observations.

Finally, a plausible downside to letting the software development practitioners evaluate their own systems was discovered. Considerable disagreements on severity ratings given by the practitioners and usability specialists were found in contribution 3. Practitioners consistently gave lower ratings than the specialists in 46 % of the problems. This indicates a lack of objectivity within the group of practitioners.

5.3 Research Question 3

The third research question is:

- *How do users driving usability evaluations perform with respect to the quality criteria?*

This question was answered by firstly conducting artificial settings research followed by research in natural settings. Findings in contribution 4 reveal a thoroughness of 24 % when users were applying the UCI method, while the forum- and diary-based identified 28 % and 33 % of the problems respectively. Additionally, contribution 5 showed a thoroughness of 64 % when users were given predefined task assignments while the non-task based condition had lower thoroughness of 28 %. Additionally, participants were most effective in identifying problems when they received inductive instructions only. Finally participants rated the inductive instructions significantly higher than the deductive. Contributions 4 and 5 show that users had a level of validity ranging from 77 % to 87 %. The reliability varied considerably between the users receiving predefined task assignments and the non-task based condition in contribution 5 where the former revealed a reliability of 28 % and the latter 4 %. In comparison, the three specialists had a reliability of 54 %. Thus, a lack of predefined task assignments increases variation between the usability problems identified. This demonstrates that authentic system use could be well suited for exploratory tests. However, if specific areas in an interface need to be evaluated, users should be given predefined tasks to keep them within these limits. Considering cost effectiveness, there was a comparable performance between the user driven approaches in contribution 4 of 37 - 39 min. per identified problem. In comparison, the conventional laboratory evaluation required 80 min. per identified problem.

In general, the user driven approach reveals a significantly lower thoroughness than the conventional laboratory evaluation conducted by specialists, especially with respect to the serious and cosmetic problems, which may seem disappointing. However, considering the cost effectiveness it is found that this approach required less than 50 % of the time to uncover each problem compared to the laboratory method while providing around 50 % of all critical problems.

5.4 Overall Research Question

In the following I return to my overall research question:

- *Can we provide support that enables software development practitioners and users to drive usability evaluations, and how do they perform with respect to the quality criteria?*

Gathering all findings from my studies of the developer driven and user driven approaches reveal that both software development practitioners and end users can be supported to drive usability evaluations. The developer driven approach reveals a higher level of thoroughness compared to the user driven approach where practitioners, due to the high level of domain knowledge were able to outperform usability specialists. On the other hand, my studies also indicate, that by increasing the number of users in a user driven approach and by providing predefined tasks, it provides a performance close to that of specialists. In general, the level of validity of identified problems is higher in the user driven approach. In terms of reliability, I found a similar performance between the developer driven and user driven approaches in which predefined tasks were given. However, not providing predefined tasks in the user driven approach reveals a significantly lower reliability. The level of downstream utility revealed in the developer driven approach resembles that in another organizational setting where usability practices had already been established. Measurements of downstream utility in a user driven approach are not covered in these contributions. I found considerable differences in cost effectiveness when comparing conventional laboratory evaluation with the user driven approach. The latter required less than half the time to uncover each problem while providing around 50 % of the critical problems. No measurements of cost effectiveness of the developer driven approach were covered in these studies.

In sum, the developer driven approach reveals a high level of thoroughness and downstream utility and the user driven approach has higher performance regarding validity. The level of reliability is comparable between the two approaches, while the user driven approach, as argued in the discussion section, will outperform the developer driven in terms of cost effectiveness.

5.5 Limitations

One limitation in my research is that I did not conduct an empirical study examining downstream utility of the user driven approach. Høegh et al. found that software development practitioners' awareness of usability problems was increased after observing user based evaluations [29]. In the user driven approach software development practitioners do not get this kind of first hand insight of the users during interaction. Thus, arguably, this awareness is compromised when applying user driven evaluations, which in turn would lead to a low level of downstream utility. Additionally, I did not conduct a study of the cost effectiveness of the developer driven approach. Hence, the level of downstream utility and cost effectiveness for the user driven and developer driven approaches remain speculative. Another limitation is that I did not evaluate the quality of the problem descriptions in case of the user driven approach.

As mentioned in the introduction of this thesis, the user driven approach is closely related to the idea of crowdsourcing. Doan and colleagues emphasize the challenge of recruiting and retaining users in relation to crowdsourcing [12]. In my studies the participants in contributions 4 and 5 were university students. As a Ph.D. student at the same university I had knowledge of the best communication channels to use when wanting to get in contact with students, which in this case was through the respective semester secretaries. Such knowledge, however, may not always be available. Additionally, as the students came from the same university as me, this could have had an effect on retaining the users, e.g. they could have been motivated to participate as we were part of the same organization. This is not always the case in practice, which is why it could be interesting to conduct further studies of how knowledge of information channels and closeness of relationships affect recruiting and retaining users in the user driven approach.

Limited generalizability is another limitation, which especially applies to the research in relation to the developer driven approach. The trade-off when applying action case as a research method is that the outcome of change and understanding comes at the cost of prediction. The outcome presented in contribution 3 may only be limited to the particular practitioners that participated and the partnering company.

Also, no full scale action research was conducted, which is a limitation to the research contributions presented in this thesis. Multiple iterations of planning, design, action, observation and evaluation in the context of a company could have revealed whether or not the developer driven and user driven approaches could have been imposed in a sustainable manner. Although I obtained an outcome of change and understanding in the action case research effort presented in contribution 3, I do not have enough data to verify sustainability. In case of the user driven approach I applied field experiments as a research method, which promote the outcome of prediction. This, however comes at the cost of change and understanding.

Finally, contribution 3 showed that, although the software development practitioners were able to remove most usability problems in the initial version of the evaluated system, a considerable amount of new problems were identified in the second version. As argued by Nielsen, it is typical that new problems occur in a new interface design [43], but this could arguably be reduced if the practitioners had received training in interaction design. As Wixon points out, then it is equally important to tell the practitioners what to do and not just what is wrong in a user interface [52].

5.1 Future Work

As a continuation of my work presented in this thesis, it would be relevant to conduct empirically based studies of downstream utility when applying the user driven approach, but also studies of the cost effectiveness of the developer driven approach. This would lead to more complete insights in these approaches with respect to the five quality criteria proposed in [19]. Although this is not part of five criteria, I still find it relevant to consider studying the quality of problem descriptions made by users in the user driven approach. Additionally, it would be relevant to further support practitioners through training in interaction design and then study how this affect the impact of usability evaluations, e.g. downstream utility. It would also be relevant to conduct field experiments with more practitioners and companies participating to increase generalizability of findings in relation to the developer driven approach. And, finally, I find a need for studying sustainability of the developer driven approach imposed in the partnering company, which could be accomplished through action research. This also applies in case of the user driven approach. At the time of writing, a 10 month period has passed without any research activities in the partnering company applying the developer driven approach. In that period the practitioners have initiated three usability evaluations on their own, which indicates some form of sustainability.

6 References

1. Ardito C., Buono P., Caivano D., Costabile M.F., Lanzilotti R., Bruun A. and Stage J. Usability Evaluation: A Survey of Software Development Organizations. In *Proceedings of SEKE*. Knowledge Systems Institute Graduate School, Skokie, IL, USA (2011).
2. Bak, J. O., Nguyen, K., Risgaard, P. and Stage, J. Obstacles to usability evaluation in practice: a survey of software development organizations. In *Proc. NordiCHI*. ACM Press (2008).
3. Benbasat, I. An Analysis of Research Methodologies. In *F.W. McFarlan (ed.), The Information Systems Research Challenge*. Harvard Business School Press, Harvard, USA (1985).
4. Bonnardel, N., Lanzone, L. and Sumner, T. Designing web sites: The cognitive processes of lay-designers. *Cognitive Science Quarterly 3*, 1 (2003), 25-56.
5. Bosenick, T., Kehr, S., Kühn, M. and Nufer, S. Remote usability tests: an extension of the usability toolbox for online-shops. In *Proc. UAHCI*, Springer-Verlag (2007), 392-398.
6. Braa, K. and Vidgen, R. Action Case: Exploring the Middle Kingdom in IS Research. In *Proc. of the Third Decenial Conference on Computers in Context*.
7. Braa, K. and Vidgen, R. Interpretation, Intervention, and Reduction in the Organizational Laboratory: A Framework for In-Context Information Systems Research. In *Accounting, Management and Information Technologies 9*, pp. 25-47. Elsevier (1999).
8. Bruce, M. and Morris, B. Managing external design professionals in the product development process. In *Technovation 14*, 9, pp. 585-599. Elsevier Science Ltd. (1994).
9. Capra, M.G. *Usability Problem Description and the Evaluator Effect in Usability Testing*. Virginia Polytechnic Institute & State University, Blacksburg (2006).
10. Castillo, J. C., Hartson, H. R. and Hix, D. Remote usability evaluation: Can users report their own critical incidents? In *proc. CHI*, 253-254. ACM Press (1998).
11. Daqing, Z., Unschuld, P.U.: China's barefoot doctor: Past, present, and future. In *The Lancet 372*, 9653, pp. 1865–1867. Elsevier Ltd. (2008).
12. Doan, A., Ramakrishnan, R. and Halevy, A.Y. Crowdsourcing systems on the World-Wide Web. In *Communications of the ACM 54*, 4, pp. 86-96. ACM, New York, USA (2011).
13. European Commissions definition of Small and Medium-sized Enterprises: http://ec.europa.eu/enterprise/policies/sme/facts-figures-analysis/sme-definition/index_en.htm.
14. Følstad, A., Hornbæk, K. Work-domain knowledge in usability evaluation: Experiences with Cooperative Usability Testing. In *J. of Systems and Software 83*, 11, 2019-2030. Elsevier Science Inc. New York, NY, USA (2010).
15. Galliers, R.D. Choosing Information Systems Research Approaches. In *Galliers, R.D. (Ed.), Information Systems Research: Issues, Methods and Practical Guidelines*. Blackwells Scientific Publications, Boston, USA.
16. Galliers, R.D. Research Issues in Information Systems. In *Journal of Information Technology 8*, pp. 92-98. Chapman Hall Ltd (1993).
17. Gray, W.D. & Salzman, M.C. Damaged Merchandise? A Review of Experiments that Compare Usability Evaluation Methods. In *Human-Computer Interaction 13*, pp. 203-262 (1998).
18. Gulliksen, J., Boivie, I., Persson, J., Hektor, A. and Herulf, L. Making a difference: a survey of the usability profession in Sweden. In *Proc. NordiCHI.* ACM Press (2004).
19. Hartson, H.R., Andre, T.S. and Williges, R.C. Criteria for Evaluating Usability Evaluation Methods. In *Int. J. of Human-Computer Interaction 13*, 4, pp. 373-410. Taylor & Francis (2001).
20. Hayes, G. The relationship of action research to human-computer interaction. In *ACM Transactions on Computer-Human Interaction (TOCHI) 18*, 3. ACM, New York, USA (2012).
21. Häkli, A. *Introducing user-centered design in a small-size software development organization*. Helsinki University of Technology (2005).

22. Hertzum, M. Problem Prioritization in Usability Evaluation: From Severity Assessments Toward Impact on Design. In *Int. J. of Human-Computer Interaction 21*, 2, 125-146. Taylor & Francis (2006).
23. Hertzum, M. and Jacobsen, N.E. The Evaluator Effect: A Chilling Fact About Usability Evaluation Methods. In *Int. Journal of Human-Computer Interaction 13*, 4, pp. 421-443- Taylor Francis (2001).
24. Howard, M., Vidgen, R., Powell, P. and Powell, J. Exploring the use of QPID: A collaborative study of B2B in the automotive industry. In *Omega 35*, 4, pp. 451-464. Elsevier (2007).
25. Howarth, J. *Supporting novice usability practitioners with usability engineering tools*. Virginia Polytechnic Institute & State University Blacksburg, Blacksburg, (2007).
26. Howarth, J., Andre, T. S. and Hartson, R. A Structured Process for Transforming Usability Data into Usability Information. *Journal of Usability Studies 3*, 1, 7-23 (2007).
27. Hwang, W. and Salvendy, G. Number of people required for usability evaluation: the 10±2 rule. In *Communications of the ACM 53*, 5, 130-133. ACM (2010).
28. Høegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B. and Stage, J. A Qualitative Study of Feedback from Usability Evaluation to Interaction Design: Are Usability Reports Any Good? In *International Journal of Human-Computer Interaction 21*, 2, pp. 173-196. Lawrence Erlbaum Associates (2006).
29. Høegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B. and Stage, J. The Impact of Usability Reports and User Test Observations on Developers' Understanding of Usability Data: An Exploratory Study. In *Int. Journal of Human-Computer Interaction 21*, 2, pp. 173-196. Taylor & Francis (2006).
30. ISO 9241-11. *Ergonomic requirements for office work with visual display terminals (CDTs) part 11: Guidance on Usability*. International Organization for Stadardization, Switzerland (1998).
31. Juristo, N., Moreno, AM and Sanchez-Segura, M. I. Guidelines for eliciting usability functionalities. In *IEEE Transactions on Software Engineering 33*, 11, pp. 744-758. IEEE Computer Society Press (2007).
32. Kjeldskov, J., Skov, M. B. and Stage, J. Instant data analysis: conducting usability evaluations in a day. In *Proc. NordiCHI*. ACM Press (2004).
33. Koutsabasis, P., Spyrou, T., Darzentas, J.S., Darzentas J.: On the Performance of Novice Evaluators in Usability Evaluations. In *Proc. PCI* (2007).
34. Law, E. Evaluating the downstream utility of user tests and examining the developer effect: A case study. In *Int. J. Human-Comput. Interact. 21*, 2, pp. 147--172. Taylor & Francis, London, England (2006).
35. Lindgaard, G. and Chattratichart, J. Usability Testing: What Have We Overlooked? In *proc. CHI.* ACM Press (2007).
36. Lund, A.M. Damaged Merchandise? Comments on Shopping at Outlet Malls. In *Human-Computer Interaction 13*, pp. 276-281 (1998).
37. McGrath, J. Dilemmatics: The Study of Research Choices and Dilemmas. In *J. McGrath, J. Martin and R. Kulka (Eds.), Judgement Calls in Research*. Sage, Beverly Hills, USA (1982).
38. McKay, J. & Marshall, P. The dual imperatives of action research. In *Information Technology & People 14*, 1, pp. 46-59. MCB UP Ltd (2001).
39. Medlock, M.C., Wixon, D., Terrano, M., Romero, R., Fulton, B. (2002). Using the RITE method to improve products: a definition and a case study. In *proc. Usability Professionals Association* (2002).
40. Metzker, E., Offergeld, M. An Interdisciplinary Approach for Successfully Integrating Human-Centered Design Methods Into Development Processes Practiced by Industrial Software Development Organizations. In *Proc. IFIP International Conference on Engineering for Human-Computer Interaction.* Springer-Verlag (2001).
41. Molich, R. *Usable Web Design*. Nyt Teknisk Forlag (2007).
42. Nielsen, J. Finding usability problems through heuristic evaluation. In *Proc. CHI*. ACM Press (1992).
43. Nielsen, J. Iterative User-Interface Design. In *Computer 26*, 11, pp. 32-41. IEEE (1993).
44. Nielsen, J. Usability inspection methods. In *Proc. CHI.* ACM Press (1994).
45. Radle, K., Young, S. Partnering usability with development: how three organizations succeeded. In *IEEE Software 18*, 1, pp.38-45. IEEE (2001).

46.Rosenbaum, S., Rohn, J. A. and Humburg, J. A toolkit for strategic usability: results from workshops, panels, and surveys. In *Proc. CHI*. ACM Press (2000).

47.Rubin, J., Chisnell, D.: *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*, 2nd. Edition. John Wiley & Sons, Inc., Indianapolis, USA (2008).

48.Sawyer, P., Flanders, A. and Wixon, D. Making A Difference – The Impact of Inspections. In *Proc. CHI.* ACM Press, New York, USA (1996).

49.Schaffer, E. *Institutionalization of Usability: A Step-By-Step Guide*. Addison Wesley, Redwood City, CA, USA, 2007.

50.Scholtz, J., Laskowski, S. and Downey, L. Developing usability tools and techniques for designing and testing web sites. In *proc. Conference on Human-Factors & the Web* (1998).

51.Skov, M. B., Stage, J. Supporting Problem Identification in Usability Evaluations. In *Proc. OZCHI*. ACM (2005).

52. Wixon, D. Evaluating usability methods: why the current literature fails the practitioner. In *Interactions 10*, 4, 28-34. ACM New York, NY, USA (2003).

53.Wynekoop, J.L. and Conger, S.A. A Review of Computer Aided Software Engineering Research Methods. In *H-E. Nissen, H. K. Klein, & R. Hirschheim (Eds.), Information systems research: Contemporary approaches and emergent traditions*. Elsevier Science Publishers, 1990.

7 Appendix A – Paper Contributions

This appendix contains the five paper contributions in their full versions. The papers have been published as follows:

[1] Bruun, A. Training Software Developers in Usability Engineering: A Literature Review. In *Proceedings of NordiCHI*. ACM Press, New York, NY, USA (2010).

[2] Bruun, A. and Stage, J. 2011. Training Software Development Practitioners in Usability Evaluations: An Exploratory Study of Cross Pollination. In *Proceedings of the 5th Workshop on Software and Usability Engineering Cross-Pollination: Patterns, Usability and User Experience (PUX 2011)*.

[3] Short version: Bruun, A. and Stage J. Overcoming the Developer Mindset Barrier towards Usability Evaluations. In *Proceedings of the 35th Information Systems Research Seminar in Scandinavia (IRIS 2012)*.
Extended version: Bruun, A. and Stage, J. Training Software Development Practitioners in Usability Testing: An Assessment Acceptance and Prioritization. Accepted for publication in *Proceedings of the 23rd Australian Computer-Human Interaction Conference (OzCHI)*. ACM Press, New York, NY (2012).

[4] Bruun, A., Gull, P., Hofmeister, L. and Stage, J., 2009. Let Your Users Do the Testing: A Comparison of Three Remote Asynchronous Usability Testing Methods. In *Proceedings of the 27th international conference on Human factors in computing systems (CHI)*. ACM Press, New York, NY, USA (2009).

[5] Bruun, A. and Stage, J., 2012. The Effect of Task Assignments and Instruction Types on Remote Asynchronous Usability Testing. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems (CHI)*. ACM Press, New York, NY, USA (2012).

Training Software Developers in Usability Engineering: A Literature Review

Anders Bruun
Aalborg University, Department of Computer Science
Selma Lagerlöfs Vej 300, DK-9220, Aalborg Oest, Denmark
bruun@cs.aau.dk

ABSTRACT
Software companies focusing on Usability Engineering face two major challenges, the first being the sheer lack of usability specialists leading to missing competences in the industry and the second, which regards small companies suffering from the constraint of low budgets, thus not being able to fund usability specialists or comprehensive consultancy. Training of non-usability personnel in critical usability engineering methods has the potential of easing these challenges. It is, however, unknown how much and what kind of research that has been committed to novice training in UE methods. This paper presents a comprehensive literature study of research conducted in this area, where 129 papers are analyzed in terms of research focus, empirical basis, types of training participants and training costs. Findings show a need for further empirical research regarding long term effects of training, training costs and training in user based evaluation methods.

Author Keywords
Training, developers, usability engineering, literature review.

ACM Classification Keywords
H5.2 Information interfaces and presentation (e.g.,HCI): User Interfaces -- User-centered design, Training, help and documentation.

INTRODUCTION

During the last decade large software companies have increased their focus on introducing usability engineering (UE) methods into the development processes. One challenge for these companies is the sheer lack of usability specialists in the industry, which leads to missing competences across the board and hence, problems with incorporating UE in the development processes [16, 24].

NordiCHI 2010, October 16–20, 2010, Reykjavik, Iceland.
 ISBN: 978-1-60558-934-3...$5.00.

However, small companies do not even have the privilege of staffing usability specialists and these have to cope with issues such as the constraint of low budgets. In practice this means that small software companies do not have the funds to pay for comprehensive consultancy or staffing of usability specialists [13, 18, 28] as they are expensive to hire [23]. A survey conducted by Gulliksen et al. acknowledges this by showing that usability specialists are primarily employed by medium-sized or large companies [12]. In their study 70 % of the 194 respondents, are employed usability specialists in companies with at least 50 employees, which are categorized as medium-sized [9]. This is also supported by Rosenbaum et al. that shows that most usability specialists are employed in large companies consisting of 250 or more employees [25].

The fact that small companies usually do not have staff with usability competences is expressed as one of the main barriers for incorporating UE in the development process [22]. This is also supported in the survey presented in [25] that shows that 17.3 % of 134 respondents mentioned missing competences as one of the main obstacles.

One way of solving the problems of incorporating UE in software companies is to increase knowledge of the subject across the set of stakeholders [12]. Thus, training non-usability personnel in critical usability engineering methods has the potential of easing problems regarding the lack of usability specialists in the industry, which is experienced in large companies. Training also provides an opportunity for small companies to apply UE methods as the developers themselves are driving the UE process, thus lessening the need to staff usability specialists, which cannot be funded.

Some studies provide promising insights regarding training of usability novices in UE methods. Metzker and Offergeld describe a software project in which developers participated in contextual task analysis, which motivated the participants to produce components with a high level of usability [22]. Another study presented in [17] describes how a non-usability specialist learned to apply a usability evaluation method. The participant stated that only little experience with cognitive psychology was needed in order to apply the method. Seffah et al. describes the opposite case in which usability specialists having sufficient technical knowhow eased integration of UE methods in the development process and gained wide acceptance from developers [29].

A reason for this could be that their perspectives are more closely aligned with that of the developers.

Although seeming to be a promising approach for solving the above mentioned problems, it is still unknown how much and what kind of research that has been committed to novice training in UE methods. This paper presents a comprehensive literature study focusing on what research has been conducted in the area of training novices in UE methods.

The following section describes the method used for identifying relevant research papers. Then an overview of the identified papers will be presented and analyzed in terms of research focus, empirical basis, types of training participants and cost. After this the findings are discussed according to future research needs and the final section presents conclusions.

METHOD

This section describes the strategy undertaken to cover the body of knowledge regarding training of novices in UE methods. The overall strategy was based on the iterative process of identifying and reading papers, then identifying and reading potentially relevant papers from references or citations.

Selection of papers - criteria

Papers were selected as relevant if they described or focused on training of novices in UE methods. Here "novices" are defined according to Bonnardel et al.'s definition:

"Lay-designers are people with little or no formal training in either web design specifically or its attendant skills (e.g., database design, graphic design, user interface design, etc.)" [5].

This definition does not, however, define what "little training" means exactly. For this the definition of Howarth et al. is used:

"Additionally, all the evaluators selected for the study had less than one year of job experience related to usability engineering, thereby qualifying them as novices" [15].

Thus, this paper defines novices as persons with less than one year of job experience and no formal training in Usability Engineering methods.

Conference and journal papers were selected as relevant sources as these have gone through scientific peer reviews in order to be accepted at conference proceedings or journals. Ph.D. and master thesis's were also included as they have been reviewed at higher exams. Books were not considered relevant in this study since it is uncertain whether or not these have gone through the same level of review. Table 2 shows how the selected papers are distributed according to publication type.

Papers were not selected if they had a purely pedagogical focus and did not describe training in particular UE methods or rendered too few details to uncover whether the training regarded analysis, design or evaluation methods. This was also the case if papers provided descriptions of particular methods aimed at novices or focused on skills needed by novices but no training was mentioned or given, for instance if novices were used as participants in experiments but details of their training was left unmentioned. Another constraint was if usability experts acted as participants or the methods used were dependent on expert knowledge, in which case the papers were excluded. The same goes for papers describing surveys of current use in software companies and papers focusing on description of courses aimed at UE lecturers. The final constraint regarded cases in which the same study by the same authors was described in different papers. In this case only one of these was chosen where journal versions were prioritized over conference papers, otherwise the latest version would be selected for inclusion.

Identification process

In order to identify the first set of relevant papers a preliminary screening was performed. This was done using Google Scholar, as this search engine covers scientific papers from a broad set of publishers and proceedings. The search criteria were based on a full-text search in which all the words "training", "developers" and "usability" were required and resulted in 33,800 records (search conducted December 1st 2009). As it would be too tedious a task to read abstracts from all these papers the first 200 abstracts were read of which 23 potentially relevant papers were selected and read in full. Eight papers were selected as relevant and defined the result of the screening process.

Papers referenced in the selected 8 papers from the screening were marked as potentially relevant. In addition the relevant 8 papers were looked up on Google Scholar which provides a utility to identify which papers are citing these. All citations were also marked as potentially relevant. Subsequently abstracts from all referenced and cited papers were read and papers fitting the selection criteria were read in full. This process continued until closure was reached after 8 iterations.

Table 1 presents the statistics from the overall reading process and shows that 4155 abstracts were read and that 286 papers were read in full ending up with 129 actually relevant papers. Note that because of the length of the references these 129 papers are listed on a website [1] and are cited using parentheses and not square brackets, e.g. "(46)".

Table 2 presents an overview of the 129 papers distributed according to publication type. From the table it can be seen that the level of quality of the majority of papers adhere to conference proceedings and journal standards.

	Potentially relevant	Read in full	Actually relevant
Screening	200	23	8
References / Citations	3955	263	121
Total	**4155**	**286**	**129**

Table 1: Statistics from the overall reading process.

Conference proceeding	73
Journal	42
Workshop	7
Thesis	7
Total	**129**

Table 2: Distribution of papers according to publication type.

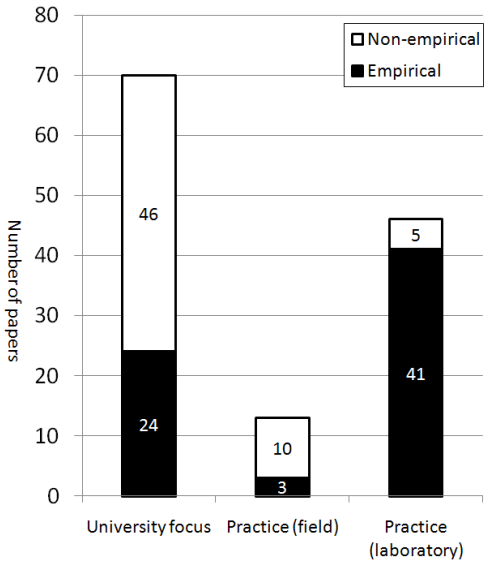


Figure 1: Number of papers distributed according to research focus and empirical basis (n = 129).

OVERVIEW OF CURRENT RESEARCH

This section provides an overview of the body of research focusing on training of novices in UE methods.

Research focus

By reading the research questions of the individual papers 3 categories of research focus were identified and are described in further detail below. Figure 1 provides an overview of the 129 relevant papers categorized according to their research focus and empirical basis.

University Focus

Seventy papers (54 %) have a university focus in which the main purpose is description, development or evaluation of UE contents for university curricula or pedagogical approaches applied in teaching UE methods to university students. The identified pedagogical papers do not focus on the curriculum contents, but on the way contents are taught, which differentiates this set of papers from those considering contents. Twenty-four are empirical studies of which 14 describe and evaluate courses with respect to student performance and perceived usefulness (6, 8, 9, 14, 24, 25, 55, 65, 90, 91, 93, 94, 120, 123). Ten are pedagogical papers evaluating the comprehensibility or effectiveness of the approach (15, 31, 35, 57, 58, 60, 83, 92, 98, 100), of which 2 are comparative studies of training materials comparing the effect of varying training conditions (15, 31). The remaining 46 papers are non-empirical of which 39 describe development of new curricula contents and course outlines (10, 11, 13, 17, 18, 20, 23, 29, 32, 33, 39, 41, 72, 74, 76, 78, 79, 80, 82, 84, 87, 89, 95, 96, 97, 99, 101, 102, 103, 105, 108, 113, 117, 118, 119, 121, 122, 125, 126, 128). Seven are studies describing new or previously applied pedagogical methods and brief descriptions of lessons learned (7, 34, 37, 38, 56, 69, 71).

Practice – Field studies

The remaining categories of research focus consist of 59 papers with the purpose of describing, developing or evaluating UE methods for use in industrial practice. These are labeled as "Practice" in Figure 1 and are divided in 2 categories. Thirteen papers (10 %) have a field focus in which institutionalization of UE methods is done through practitioner training while considering the organizational context. These papers consider company size, staffing and development process and 3 are empirical studies where papers (1) and (36) present studies of integrating UE methods into small-sized companies through developer training and (66) uses a large company as their case. The remaining 10 papers describe non-empirical research by presenting theoretical frameworks for integration of various UE methods into development processes and present suggestions for industry training programs (12, 26, 27, 28, 52, 54, 67, 81, 86, 104).

Practice – Laboratory studies

Forty-six papers (36 %) focus on description, development or evaluation of UE methods or tools isolated from the organizational context and is thus labeled as "laboratory". Six of these papers conduct empirical studies of novice performance when applying UE methods of which 5 present comparative studies of differences between usability specialists and novices (3, 85, 109, 111, 129). The sixth paper (46) evaluates the comprehensibility of a novel UE method when used by novices without comparing novices and specialists. Nine of the 46 "laboratory" papers focus on development or descriptions of software tools to support novices in applying UE heuristics or methods. Four of these are empirical studies that describe development of tools and evaluation of these using novices (50, 70, 106, 115) and 5

are non-empirical describing the development of specific software applications to support novices (4, 42, 59, 75, 114). Seventeen of the 46 "laboratory" papers have the main purpose of comparing two or more UE methods empirically using novices (2, 19, 21, 22, 30, 43, 44, 48, 49, 53, 61, 62, 63, 64, 68, 77, 107), while 14 papers are empirical evaluations of single methods applied by novices (5, 16, 40, 45, 47, 51, 73, 88, 110, 112, 113, 116, 124, 127).

Summary – Research focus

The above observations show that the majority of papers focus on development, description and evaluation of university curricula content or pedagogical approaches (54 %) and UE methods in isolation from organizational context (36 %). Few of the identified papers (10 %) focus on institutionalization issues in the field by considering the organizational context and that only 3 of these (2 %) are empirical studies on the subject. As this paper seeks to identify the body of research regarding training of novices in UE methods targeted for use in industrial practice, the remaining sections focus on the papers doing so, i.e. the 59 papers in the "Practice" categories shown in Figure 1.

Participant types

Table 3 presents an overview of the 59 papers distributed according to their empirical basis and training participants. The matrix is divided in columns representing the empirical basis of the relevant papers, i.e. non-empirical and empirical, where empirical studies may base findings on experiments using either students, practitioners or both as subjects. Note that it was not possible to determine whether the subjects in the empirical studies of papers (15, 73) were students or practitioners, thus an additional category labeled "unclear" was necessary.

Empirical basis				
Non-empirical	**Empirical**			
	Students	Practitioners	Both	Unclear
4, 12, 26, 27, 28, 42, 52, 59, 67, 81, 104, 114	2, 5, 16, 19, 21, 30, 31, 35, 40, 43, 44, 45, 46, 49, 50, 51, 61, 62, 63, 64, 68, 77, 85, 106, 107, 109, 110, 111, 112, 113, 127, 129	1, 3, 22, 36, 47, 48, 53, 54, 66, 70, 86, 115, 124	88	15, 73
n = 12	**n = 32**	**n = 13**	**n = 1**	**n = 1**

Table 3: Papers distributed according to empirical basis and participant type (n = 59).

Non-empirical

The table shows that 12 papers of the 59 (20 %) describe non-empirical research targeted at practitioners. The majority of these discuss field issues of institutionalizing UE into software companies and the remaining describes development of tools to support novices.

Students

Thirty-two papers (54 %) are empirically based using students for observations on training in methods targeted for use by practitioners in isolation from the organizational context, i.e. laboratory experiments. Most of these papers concern evaluation of a single or multiple UE methods and a few regard studies of pedagogical approaches and tool support.

Practitioners

Thirteen papers (22 %) describe empirical studies based on observations of novice practitioners trained in UE methods, which are also aimed for use by practitioners. The papers described in this set have a field focus on institutionalization via training of practitioners and others evaluate UE methods and supporting tools in laboratory conditions isolated from organizational contexts.

Both

A single paper (2 %) uses students and practitioners as the empirical basis. That paper describes a comparative laboratory study where the purpose is to evaluate a UE method using 4 different experimental conditions, of which 1 includes practitioners and the others include students.

Summary – Participant types

Summarizing on the above it is shown that most of the current empirical research focusing on training in UE methods, for use by practitioners, is actually done using students as the empirical basis. Additionally, few of the identified empirical studies report of training novice practitioners in UE methods targeted for use in industry practice. Another observation is the fact that only 1 paper includes students and practitioners in the same study.

Training contents

Figure 2 provides an overview of the UE methods which are described and taught in the 59 papers targeting practitioners. For simplicity the methods are categorized according to the usability engineering activities in which they are to be applied [26]. The first activity category is "analysis", which covers survey or task analysis methods used for needs analysis or user profiling. The second category is "design", which regards design principles, patterns or prototyping methods applied during creation of the user interface. The third activity category is "evaluation", which covers usability evaluation methods such as heuristic inspection or usability testing with users.

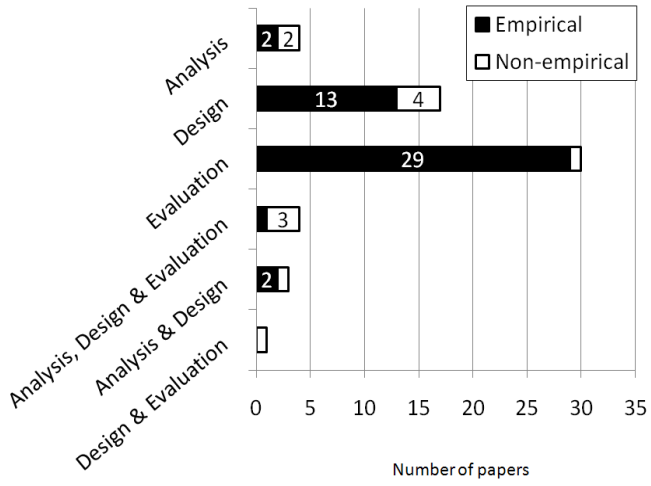


Figure 2: Number of papers targeted at practitioners distributed according to training contents (n = 59).

Figure 2 shows that 4 papers report on training in analysis methods exclusively, where 2 are empirical studies (1, 51) and 2 are non-empirical (52, 104). It can also be seen that 16 papers focus on training in design methods only, of which 4 are non-empirical (4, 42, 59, 114) and 13 are empirical (3, 16, 19, 31, 35, 61, 62, 66, 70, 86, 106, 115). The majority of papers focus on training novices in evaluation methods solely where a single paper is non-empirical (67) and 29 are based on empirical observations (2, 5, 21, 22, 30, 40, 43, 44, 45, 47, 48, 49, 50, 53, 63, 64, 68, 77, 85, 88, 107, 109, 110, 111, 112, 113, 124, 127, 129).

The remaining 8 papers describe training in several categories of methods where 3 are empirical (46, 54, 36) and 5 are non-empirical (26, 27, 28, 81, 12). Summarizing Figure 2 it can be seen that 11 papers describe training in analysis methods, 24 in design methods and 35 in evaluation methods targeted for use by practitioners.

Training costs

This section provides an overview of the reported training costs in the 59 papers targeting practitioners of which 21 papers (36 %) mention training costs as a focal point.

Training costs and field studies

Figure 3 provides an overview showing that 6 of these 21 papers consider the organizational context, thus having a field focus on institutionalization. The 2 empirical studies are using practitioners as the empirical basis. One of these 2 papers train participants in analysis methods (1) while the other describes training in analysis, design and evaluation methods (36). The remaining 4 papers having this research focus are non-empirical and present theoretical frameworks for integration of various UE methods into development processes. Two of these recommend training in several methods (27, 28) while the final 2 suggest training in design or evaluation methods exclusively (67, 86).

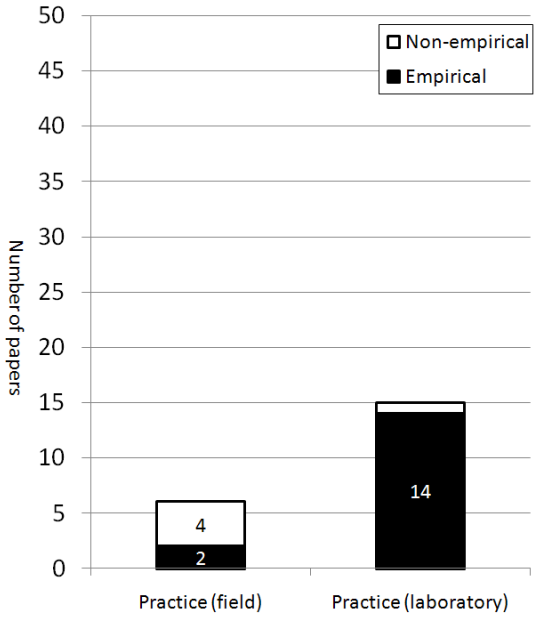


Figure 3: Number of papers focusing on training cost distributed according to research focus (n = 21).

Training costs and laboratory studies

Fifteen of the papers considering training costs have description, development or evaluation of UE methods or supporting tools as a focal point. None of these consider organizational contexts, thus being categorized as laboratory studies. One of the 15 papers is a non-empirical study and describes the development of a tool to support novice practitioners in design tasks (42). The last 14 are empirically based of which 5 use practitioners as participants and 1 uses students and practitioners. The papers evaluate novice use of UE methods and supporting tools, where training contents regard either evaluation or design methods exclusively (47, 48, 53, 70, 88, 115). The remaining 8 papers use students as the empirical basis of which 1 describe development and evaluation of a tool to support practitioners in design activities (106) and the final 7 compare evaluation methods (21, 45, 68, 77, 85, 109, 127).

Differences in training hours

Fifteen of the 59 papers mention training costs in number of hours as the measuring unit of which 7 focus on cost (21, 36, 45, 47, 48, 53, 109) and 8 do not (2, 5, 30, 40, 49, 62, 107, 111). Figure 4 shows a box plot of these 15 papers distributed according to their focus on cost and training duration in hours.

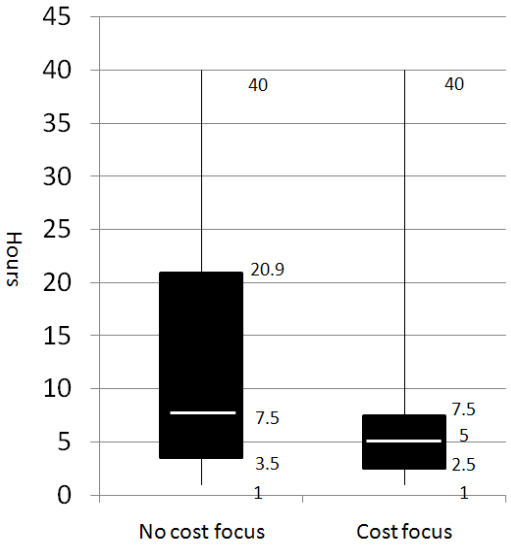


Figure 4: Box plot of empirical based papers targeted at practitioners (n = 15 papers mention training cost in hours).

The first quartiles (0 – 25 %) show a lower bound of 1 – 3.5 hours for papers not focusing on training cost where the 2 papers contained herein focus on training students in usability evaluation methods exclusively (2, 40). In case of cost focus the lower bound is between 1 – 2.5 hours and includes 2 papers describing training novice students (45) and practitioners in evaluation methods solely (53).

The second quartile (25 – 50 %) shows that the number of training hours is between 3.5 – 7.5 and 2.5 – 5 hours for non-cost and cost focus respectively. Papers (5, 107) do not focus on training costs and report on training students in evaluation methods exclusively, which is also the case for paper (21) focusing on cost.

The third quartiles show larger differences spanning from 7.5 – 20.9 hours for papers not having training costs as a focal point and 5 – 7.5 for papers that do. These 2 sets contain 4 papers which describe training of novice practitioners focusing on cost (47, 48) and training of students with no cost focus (30, 49). The 4 papers all describe training in evaluation methods solely.

The fourth quartiles span from 20.9 – 40 hours in the case of papers without a training cost focus and 7.5 – 40 for the papers mentioning low training costs. The 2 papers not mentioning costs describe educating students in design or evaluation methods solely (62, 111). One of the 2 papers considering low training costs educate novice students in evaluation methods exclusively (109) and the other train novice practitioners in all aspects of UE, i.e. analysis, design and evaluation methods (36).

Thus, from Figure 4 it can be seen that papers focusing on cost overall spend less hours training participants in UE methods. This is in part indicated by the upper bounds of quartiles 1, 2 and 3, which are lower for papers having a cost focus. Another, less apparent, indicator is the medians, which are 7.5 for papers with a non-cost focus and 5 for those considering cost. However, these indications are inconclusive due to the few papers describing training costs in hours and due to the overlaps in training hours between papers focusing on cost and those that do not, especially in the first and second quartiles. The fact that there are considerable variations in the number of hours, even if training contents are similar, is also adding to the inconclusiveness. Thus, the needed amount of training in, e.g. evaluation methods is still an open question.

Summary – Training costs

In sum only 2 empirical studies focus on institutionalization and organizational contexts in conjunction with training costs. Figure 4 also indicate disagreements on what constitutes a necessary amount of training to educate novices in UE methods, even in the first 3 quartiles, in which all papers describe similar training contents (evaluation methods exclusively). Variations in the fourth quartiles may be explained through differences in training contents, i.e. one paper covers analysis, design and evaluation where others consider evaluation or design methods only.

Evaluation Methods

This section focuses on papers reporting of training in usability evaluation methods. Methods related to analysis and design activities are of course also important training areas, however, results from usability evaluation methods have proven to be effective in creating the wake-up calls necessary for companies to start focusing on UE or to increase the awareness of developers [14, 27], which is why this section is dedicated to evaluation methods. From Figure 2 it can be extracted that 30 of the 59 papers (51 %) describe empirical studies of evaluation methods.

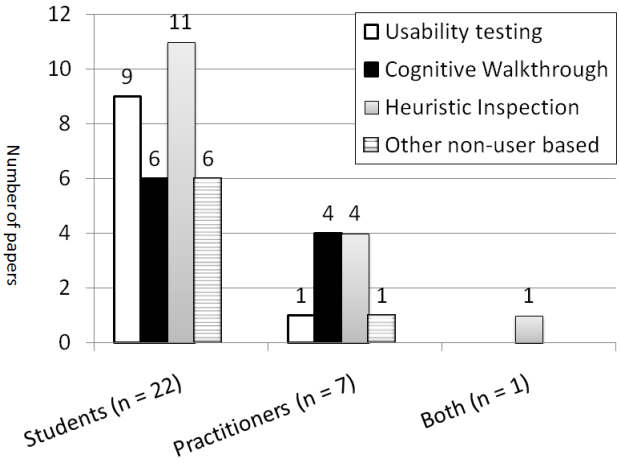


Figure 5: Number of empirical studies targeted for use in practice distributed according to participant type (n = 30).

Figure 5 provides an overview of the specific evaluation methods taught in the 30 papers distributed according to the participant types. Paper (88) is represented as "both" as the empirical base consists of students and practitioners.

Usability testing

Ten of the 30 papers report of training participants in user based methods and are mentioned as either think-aloud, usability test or user testing in the various studies. Taken together these are labeled as usability testing. Figure 5 shows that a single paper report on training in usability testing using novice practitioners as the empirical base (36) and 9 papers report on student training in usability testing (2, 30, 45, 50, 64, 109, 110, 111, 127).

Non-user based evaluation methods

A total of 26 papers describe training in non-user based methods. Ten of these consider training in Cognitive Walkthrough (CW) where 4 are based on observations of novice practitioners (22, 47, 48, 124) and 6 on students (30, 40, 43, 49, 64, 107). The majority of papers report using the Heuristic Inspection (HI) method of which 4 are based on training of practitioners (22, 36, 53, 88) and 10 on students (21, 30, 44, 63, 68, 77, 85, 88, 112, 113). Seven papers describe various non-user based methods such as Metaphors of Human Thinking, Barrier Walkthrough, Abstract Tasks or Programmable User Model, which are categorized collectively as "other non-user based" evaluation methods in Figure 5. This figure shows a single paper describing training of practitioners in a method from the "other non-user based" category method (66) while the 6 papers are student based (2, 5, 21, 43, 44, 129).

Summary – Evaluation methods

As mentioned in the introduction of this paper resource demands is one of the main obstacles companies face when wanting to incorporate UE into the development process. For this reason it is not only important to focus on training costs but also the costs of using the particular methods. Considering the 30 papers describing training of usability evaluation methods, 26 (87 %) report of training in the Cognitive Walkthrough, Heuristic Inspection and other non-user based methods, all of which are considered low-cost as no laboratory or end-users are required. There also exist varieties of user based usability testing methods focusing on low-cost, i.e. Instant Data Analysis or Rapid Iterative Testing and Evaluation [20, 21]. However, none of the above 30 papers focus on training participants in a resource saving user based evaluation method.

DISCUSSION

The above observations show that the current body of literature primarily focuses on development, description or evaluation of curricula contents or pedagogical approaches aimed at university students (54 %). Another large research area is description, development or evaluation of UE methods isolated from organizational contexts (36 %). Less focus (10 %) is devoted to institutionalization issues considering the context, in which the UE methods are to be used. This, however, is a crucial area of research, as methods isolated from the real world context potentially are of little practical relevance [31]. In software companies there exist several constraints such as team buy-in, resources, change management, political environments, technology and personnel [19, 29, 31], all of which pose challenges not experienced in studies using university students as participants or evaluation of methods in isolation. Three of the papers focusing on institutionalization (2 %) present empirical studies of integrating UE methods into companies through developer training (1, 36, 66). Results from all 3 studies acknowledge the potential of training novice practitioners in UE methods. Post-project interviews conducted in (1) revealed that participants perceived the training as being satisfying and important to future usability work, an observation similar to the findings described in (36), where the 13 participants graded the training course as 4.25 on a 1 – 5 scale (5 = very good). (66) report of lessons learned statements which indicates that 80 % liked the training course and 20 % did not and (1) and (36) indicate similar reactions where the practitioners wanted more training in UE. (36) reveals more detailed results by showing that the novice practitioners, on the other hand, experienced problems in categorizing the severity of identified usability problems when asked to conduct a heuristic inspection. A design exercise conducted before the training is also compared to one conducted after and shows some minor improvements in proposed designs. Thus, these studies have shown promising short term results, however, there is still a need for empirical research of long term effects on institutionalizing UE via training of novice practitioners. This is important because we need to include and cater for various organizational constraints.

As mentioned above, it is important to focus on practitioners in industry. Research in UE methods targeted for use by practitioners is reported in 59 of the identified papers (46 %), of which 12 are non-empirical studies and 32 and 13 papers apply novice students and practitioners respectively, while a single paper uses both. Thus, current empirical research mainly focus on using students as participants, which in turn indicates a future need for empirical studies using practitioners employed in the software industry. In total 10 % of the 129 identified papers consider this type of participants. One of the reasons for the focus on using students as the empirical basis is that this approach is less demanding in terms of planning and external involvement. Nevertheless it is still important to get more focus on using practitioners as the empirical basis in order to include constraints such as team buy-in, political environments etc.

As argued in the introduction high resource demands is one of the largest obstacles of introducing UE into software companies, which especially is the case for small software companies that have insufficient funds to pay for comprehensive consultancy or staffing of usability

specialists [4, 13, 18, 25, 28]. According to Seffah et al. practitioners want to produce software with a high level of usability, but as they work under time and budget restrictions they do not want UE to consume too much of their time [29]. For this reason a considerable amount of research has been committed to developing resource saving methods, mainly in the area of usability testing [20, 21, 23]. Resource saving methods are especially relevant for small companies that do not have the funding to incorporate UE. It also implies that not only should the taught UE methods be resource saving, but this must also apply to the training itself as there is no point in teaching resource saving methods if small companies cannot afford the training. This is also supported in [16] and [30], where the latter shows that most organizations tend to view training as a cost rather than an investment. Although cheap training is important, this review shows that 21 (36 %) of the 59 papers describing methods for practitioner use, focus on training costs. Additionally, 2 of these 21 papers are empirical studies of institutionalization issues in which the organizational context and training of practitioners are considered. Results show that although papers focusing on training costs report a lower number of training hours compared to papers without cost as a focal point, there are still disagreements on what constitutes a necessary amount of training to educate novices in UE methods. In this regard there is a need for obtaining knowledge of how much training is actually needed in order to apply UE methods satisfactorily. There is also a need for further empirical research considering cost and institutionalization issues.

Usability evaluation methods have proven to be valuable in generating the wake-up calls needed to make companies focus on UE and in increasing the awareness of developers [14, 27]. Thus, evaluation methods could be the potential best way to start a series of UE training sessions. Fonseca et al. [10] and Edwards et al. [8] also describe university curricula in which they begin the courses with letting students evaluate user interfaces, as this increases their awareness of interface problems. The survey described in [25] also states that usability testing in and without a lab are the most preferred methods by software companies. Furthermore several studies indicate that user based testing is superior to inspection or walkthrough methods in this regard. User based tests provide valuable first hand observations of the problems experienced by real users, which in turn increases motivation for making adjustments to the user interface [14]. The reason for this may be located in the fact the user testing provides empirical evidence of the problems at hand compared to theoretical inspections [6]. Additionally, the study presented in [11] indicates that a majority of evaluators prefer user based methods over the inspection methods Metaphors of Human-Thinking and Cognitive Walkthrough. A similar study described in [2] shows that user based methods was rated above inspection methods regarding pleasantness of use. Thus, several studies indicate that user based testing potentially increases practitioner buy-in, which in turn may ease the institutionalization of UE, a notion supported in [7]. This study shows that 10 papers describe training in user based evaluation methods, of which 1 uses practitioners as the empirical base. Thus, there is a need for empirical studies of training novice practitioners in user based evaluation methods.

Considering the importance of user based methods and previous discussions on training costs, focus on organizational contexts and practitioners, this paper shows that a single empirical based paper is focusing on these crucial issues (36). That paper shows promising first results in integrating UE into a small software company through training in user based evaluations. The training provided positive attitudes towards UE from the developers who afterwards were further motivated in learning about usability aspects in software development. The user based method taught was, however, not reported as being resource saving, thus in part conflicting with the needed aim of introducing resource saving methods via cheap training to small companies. This also constitutes a future research need.

CONCLUSIONS AND FUTURE RESEARCH NEEDS

This paper has presented a comprehensive literature study of the research conducted in the area of training novices in Usability Engineering (UE) methods. The 129 identified papers have been analyzed in terms of research focus, empirical basis, types of training participants and training costs. Five key areas for future research needs are identified and listed below.

Firstly 13 papers (10 %) focus on institutionalization issues by considering the organizational contexts, of which 3 are empirical based studies. The 3 empirical studies in this area have focused on measurements of short term effects of introducing UE into the organizations, thus indicating a need for further empirical research of long term effects. Secondly there is also a future need for focus on empirical studies using practitioners as few of the identified papers (10 %) consider this type of training participants. Third, due to disagreements in the current body of literature, this review also shows a need for further empirical research focusing on training costs and the amount of training necessary to obtain a satisfactory level of method usage. Fourth, in addition to training costs, few (2) of the identified papers are empirical studies in which institutionalization and cost issues are considered at the same time, hereby indicating another need. Fifth, as discussed previously, user based evaluation methods seem to provide the best wake-up call for software companies, and this review shows that a single empirical based paper is focusing on training novice practitioners in such a method. This leaves room for future research in this area.

REFERENCES

1. List of selected papers analyzed in this literature review. http://www.cs.aau.dk/~bruun/references.html.
2. Ardito, C., Costabile, MF, De Angeli, A. and Lanzilotti, R. Systematic evaluation of e-learning systems: an experimental validation. In *Proc. NordiCHI 2006*, ACM Press (2006).
3. Bailey, G. S. Iterative methodology and designer training in human-computer interface design. In *Proc. INTERACT 1993*, ACM Press (1993).
4. Bak, J. O., Nguyen, K., Risgaard, P. and Stage, J. Obstacles to usability evaluation in practice: a survey of software development organizations. In *Proc. NordiCHI 2008*, ACM Press (2008).
5. Bonnardel, N., Lanzone, L. and Sumner, T. Designing web sites: The cognitive processes of lay-designers. *Cognitive Science Quarterly 3*, 1 (2003), 25-56.
6. Brown, C., Pastel, R. Combining distinct graduate and undergraduate HCI courses: an experiential and interactive approach. In *Proc. SIGCSE 2009*, ACM Press (2009).
7. Dayton, T. Skills needed by user-centered design practitioners in real software development environments: report on the CHI'92 workshop. *ACM SIGCHI Bulletin 25*, 3 (1993), 16-31. ACM Press (1993).
8. Edwards, A., Wright, P. and Petrie, H. HCI education: We are failing–why? In *Proc. HCIEd 2006*, Springer (2006). Springer (2009).
9. European Commissions definition of Small and Medium-sized Enterprises:

   http://ec.europa.eu/enterprise/policies/sme/

   facts-figures-analysis/sme-definition/index_en.htm
10. Fonseca, M., Jorge, J., Gomes, M., Gonçalves, D. and Vala, M. Conceptual design and prototyping to explore creativity. *IFIP International Federation for Information Processing 289* (2009), 203-217. Springer (2009).
11. Frøkjær, E., Hornbæk, K. Metaphors of human thinking for usability inspection and design. *TOCHI 14*, 4 (2008). ACM Press (2008).
12. Gulliksen, J., Boivie, I., Persson, J., Hektor, A. and Herulf, L. Making a difference: a survey of the usability profession in Sweden. In *Proc. NordiCHI 2004*, ACM Press (2004).
13. Häkli, A. *Introducing user-centered design in a small-size software development organization*. Helsinki University of Technology (2005).
14. Høegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B. and Stage, J. The Impact of Usability Reports and User Test Observations on Developers' Understanding of Usability Data: An Exploratory Study. *Int. Journal of Human-Computer Interaction 21*, 2 (2006), 173-196. Taylor & Francis (2006).
15. Howarth, J., Andre, T. S. and Hartson, R. A Structured Process for Transforming Usability Data into Usability Information. *Journal of Usability Studies 3*, 1, 7-23 (2007).
16. Ji, Y. G., Yun, M. H. Enhancing the minority discipline in the IT industry: A survey of usability and User-Centered design practice. *Int. Journal of Human-Computer Interaction 20*, 2 (2006), 117-134. Taylor & Francis (2006).
17. John, B.E. and Mashyna, M.M. Evaluating a Multimedia Authoring Tool with Cognitive Walkthrough and Think-Aloud User Studies. *Journal of the American Society of Information Science 48*, 9 (1995), 1004–1022.
18. Juristo, N., Moreno, AM and Sanchez-Segura, M. I. Guidelines for eliciting usability functionalities. *IEEE Transactions on Software Engineering 33*, 11 (2007), 744-758. IEEE Computer Society Press (2007).
19. Karat, C. M., Campbell, R. and Fiegel, T. Comparison of empirical testing and walkthrough methods in user interface evaluation. In *Proc. CHI 1992*, ACM Press (1992).
20. Kjeldskov, J., Skov, M. B. and Stage, J. Instant data analysis: conducting usability evaluations in a day. In *Proc. NordiCHI 2004*, ACM Press (2004).
21. Medlock, M. C., Wixon, D., Terrano, M., Romero, R. and Fulton, B. Using the RITE method to improve products: A definition and a case study. In *Usability Professional's Association Annual Conference 2002*.
22. Metzker, E., Offergeld, M. An Interdisciplinary Approach for Successfully Integrating Human-Centered Design Methods Into Development Processes Practiced by Industrial Software Development Organizations. In *Proc. IFIP International Conference on Engineering for Human-Computer Interaction 2001*, Springer-Verlag (2001).
23. Nielsen, J. Usability inspection methods. In *Proc. CHI 1994*, ACM Press (1994).
24. Nielsen, J. Finding usability problems through heuristic evaluation. In *Proc. CHI 1992*, ACM Press (1992).
25. Rosenbaum, S., Rohn, J. A. and Humburg, J. A toolkit for strategic usability: results from workshops, panels, and surveys. In *Proc. CHI 2000*, ACM Press (2000).
26. Rubin, J. *Handbook of usability testing*. John Wiley & Sons Inc., New York, NY, USA, 1994.
27. Schaffer, E. *Institutionalization of Usability: A Step-By-Step Guide*. Addison Wesley, Redwood City, CA, USA, 2007.
28. Scholtz, J., Laskowski, S. and Downey, L. Developing usability tools and techniques for designing and testing web sites. In *Proc. Conference on Human-Factors & the Web 1998*.

29. Seffah, A., Gulliksen, J. and Desmarais, M. An Introduction to Human-Centered Software Engineering. In *Seffah, A., Gulliksen, J., Desmarais, M.C. (eds.) vol. 1, 59-70*. Springer, Netherlands (2005).

30. Taylor, M. J., England, D. and Gresty, D. Knowledge for Web site development. *Internet Research 11*, 5 (2001), 451-461. MCB UP Ltd. (2001).

31. Wixon, D. Evaluating usability methods: why the current literature fails the practitioner. *Interactions 10*, 4 (2003), 28-34. ACM Press (2003).

Training Software Development Practitioners in Usability Evaluations: An Exploratory Study of Cross Pollination

Anders Bruun
and Jan Stage

Department of Computer Science, Aalborg University,
Selma Lagerlöfs Vej 300, DK-9220, Aalborg Oest, Denmark
{bruun, jans}@cs.aau.dk

Abstract. Successful integration of usability evaluation into software development processes requires software companies to employ personnel that possess skills within both usability and software development. However, the sheer lack of usability specialists and their cost are two limiting factors for software companies wanting to integrate usability evaluation. A possible solution to these problems is to cross pollinate by training existing personnel in conducting usability evaluations and analyzing the collected data. This exploratory study extends previous research by showing that it is possible to provide software development practitioners from industry with key knowledge on usability evaluation. Results show that a pair of practitioners can identify the same number of problems as one usability specialist after 14 hours of training. Furthermore, software practitioners are better at providing clear and precise problem descriptions than at describing the impact, cause, user actions and providing data support for observations.

Keywords: Usability evaluation, training, software development practitioners, problem identification, problem descriptions.

1 Introduction

For the past decade software companies have increased their focus on integrating usability engineering (UE) into development processes. A considerable challenge for these companies is the limited supply of usability specialists in the industry, which leads to integration problems caused by missing key knowledge [12], [16]. Another challenge, which especially relates to small software companies, is that these have to cope with the constraint of low budgets. In practice this means that small companies do not have the funds to pay for comprehensive consultancy or staffing of usability specialists [9], [13]. A survey conducted by Gulliksen et al. supports this by showing that usability specialists are primarily employed by medium-sized or large companies [8]. The fact that small companies usually do not have staff that possesses usability knowledge is expressed as one of the main barriers for integrating UE into software development processes [14], [17].

One way of solving these problems may be to cross pollinate disciplines by increasing usability knowledge across existing personnel, an approach which previously has provided positive results. Metzker and Offergeld for instance describe a software project in which developers participated in contextual task analysis, which motivated them to produce components with a high level of usability [14]. However, a recent literature review presented in [2] shows that the majority of related studies are applying university students as the empirical basis, which leaves room for further studies on software development practitioners' ability to apply UE methods. Another point for consideration is the fact that the majority of related work focus on measuring quantitative aspects of e.g. usability evaluations such as the number of problems identified. Thus in the case of usability evaluation, there is also a need to report findings on aspects such as the quality of problem descriptions and in particular which parts of the descriptions that software practitioners find difficult to fulfill.

This exploratory study extends previous research by studying how software development practitioners from industry perform in identifying usability problems and by providing insights in the quality of their problem descriptions. We have chosen to train practitioners in user based evaluation methods, as such methods have proven to be effective in creating the wake-up calls necessary for companies to start focusing on UE or to increase the awareness of developers [11].

The paper is structured in the following way. First we provide a description of the experimental method applied after which we present our findings and discuss these with respect to related work. Finally we provide the conclusion and point out avenues of future work.

2 Method

In this section we describe the scientific method applied which consisted of a training course that provided key usability knowledge and an evaluation experiment that assessed software practitioners' performance in analyzing usability evaluation data. We start by presenting the participants of the training course and experiment.

2.1 Participants

Software Development Practitioners. Five software development practitioners (henceforth mentioned as "SW-P" or "practitioners") employed in a small software company participated in the experiment. Table 1 shows an overview of their job functions within the company and experience with usability work in general. SW-P 1 had 1.5 years of job experience as a systems developer and did not have any experience with usability evaluation during his employment at the company. However, as part of his education he had previously participated in a HCI course and in the conduction of 4-5 usability evaluations (7 years back). SW-P 2 was a test manager with 8 years of job experience in the company and did not have any practical experience in applying usability methods. She had read a single chapter on the subject during her education. SW-P 3 had 2 years of experience as project manager and systems developer, but had no previous experience with usability work. SW-P 4 had

3.5 years of experience as a systems developer in the company and did not have any experience with usability work before this study commenced. SW-P 5 had worked as a systems developer for 2 years in the company. Additionally he had participated in a HCI course during his education and had experience from conducting a single usability evaluation 13 years back.

Table 1. Overview of the software development practitioners' (SW-P) job functions within the company and experience with usability.

SW-P no.	Function	Usability Experience
1	Systems developer	HCI course + 4-5 evaluations
2	Test manager	Through literature
3	Project manager + systems developer	None
4	Systems developer	None
5	Systems developer	HCI course + 1 evaluation

Trainers. The two authors prepared and held a usability training course for the practitioners (see course description in section 2.2 below).

External Raters. Three usability specialists acted as external raters of the problem lists produced during the evaluation experiment as we did not want to evaluate the outcome of our own training (see section 2.3 for further details). None of these raters had taken part in the training or the conduction of the usability evaluation and are thus considered to be unbiased.

Test Users. Six test users were recruited for the evaluation experiment, all of which were representative end users of the evaluated system.

2.2 Training Course

The authors conducted a two-day training course (14 hours) on user based usability evaluations. The course was held as a combination of presentation and exercises. At the end of the course we gave the practitioners a homework assignment in which they were asked to analyze five video clips from a previous usability evaluation of an e-mail client. We collected the resulting problem lists and gave the participants feedback on how they could improve their problem descriptions.

2.3 Evaluation Experiment

The emphasis of this study is based on the usability evaluation conducted by the 5 practitioners after completing the training course. Due to planning time and busy participant calendars this was executed one month later.

System. The system evaluated was a web application that citizens may use when they move from one address to another. The system was partly developed by the software company in which the 5 practitioners were employed but none of the practitioners had participated in the development of the particular system.

Setting. The evaluation was conducted in the usability laboratory at the university which consists of a test room with cameras and a microphone and an observation room behind a one way mirror. During each session a test user was sitting at a table in the test room using the web application. Next to the user a practitioner acting as test monitor would be positioned.

Procedure. All practitioners took part in planning the test while three of these (SW-P 1, 2 and 3, see Table 1) conducted the evaluation. Afterwards, all 5 analyzed the obtained video material and described the usability problems. The usability evaluation was conducted in one day where SW-P 1, 2 and 3 acted as test monitor two times each. After completing the evaluation all 5 practitioners analyzed the video material from the lab individually. One of the authors analyzed the same video material and this person is mentioned as the "HCI specialist" from this point on. The practitioners and the HCI specialist used the same template for describing problems in order to promote a consistent format. The three unbiased external raters were then asked to evaluate the quality of the problem lists created by the 5 practitioners and the HCI specialist. Finally, the HCI specialist held a meeting with the five practitioners in which the 6 individual problem lists were merged into a total list of usability problems, which served as a white list to calculate the thoroughness in identifying problems. At the same meeting a debriefing interview with each of the developers were conducted.

Analysis of Problem Description Quality. The three unbiased external raters were asked to evaluate the quality of the problem lists created by the 5 practitioners and the HCI specialist. To measure the quality of the lists, the raters were asked to first read each problem list and then provide a rating on a scale from 1 – 5 (1 = "Not fulfilled", 2 = "Scarcely fulfilled", 3 = "Partially fulfilled", 4 = "Almost fulfilled" and 5 = "Fulfilled"). These ratings were given on the following attributes (based on the research presented in [3]):

1. Be clear and precise while avoiding wordiness and jargon
2. Describe the impact and severity of the problem
3. Support your findings with data
4. Describe the cause of the problem
5. Describe observed user actions

Finally the external raters were asked to provide a qualitative assessment of each list, i.e. to provide arguments of the ratings given and examples from the problem lists.

4 Results

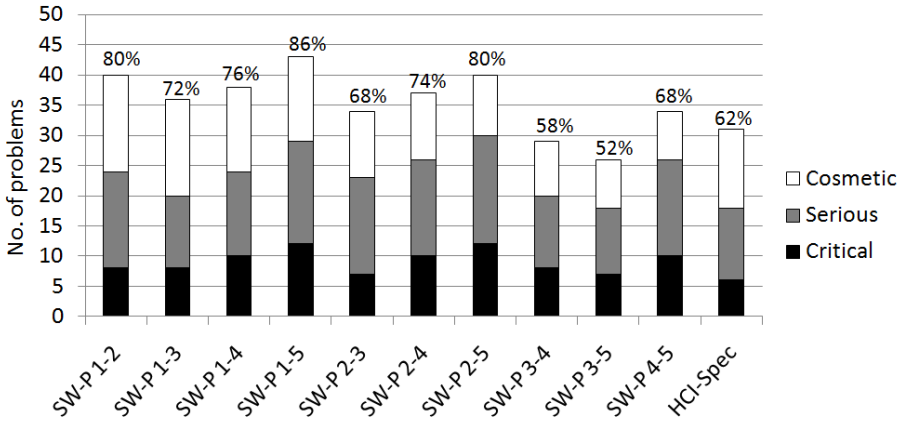
This section presents our findings and is divided in two subsections where the first describes practitioners' ability to identify problems while the second provides qualitative details on their ability to describe usability problems.

4.2 Identification of Usability Problems

Results show that a total of 50 usability problems were identified of which 12 are critical, 19 serious and 19 cosmetic, see [15] for elaboration of severity categorizations. The HCI specialist identified 31 of the problems (62 %) and the practitioners identified between 14 (28 %) and 33 (66 %), the mean being 24.2 (SD=8.1), or 48.4 %. On average practitioners identified 78 % of the problems found by the HCI specialist. Considering the amount of critical problems practitioners identified a mean of 6.8 (57 %) (SD=2.6) where the most and least thorough found 83 % (SW-P 1) and 25 % (SW-P 3) respectively. In comparison the HCI specialist identified 6 (50 %). Considering the serious problems practitioners found 10 (SD=3.9) on average (53 %), the highest being 79 % (SW-P 2) and the lowest 21 % (SW-P 3). The HCI specialist found 12 serious problems (63 %). In the case of cosmetic problems the average is 7.4 (SD=3.2), or 39 %, where SW-P 1 identified most (63 %) and SW-P 4 fewest (21 %), while the HCI specialist found 13 (68 %).

Pair wise Identification. In practice it can be too resource demanding to utilize five evaluators in analysis of usability data, thus in the following we study the effectiveness of each pair of practitioners. Figure 1 provides an overview of the number of problems identified by all pairs of practitioners. All pairs identifies an average of 35.7 (SD=5.2) of all problems (71.4 %), where SW-P 1 and SW-P 5 was the pair that identified most problems (86 %) and SW-P 3 and SW-P 5 identified fewest (52 %). In comparison, the HCI specialist identified 62 %.

Figure 1. Overview of the number of problems identified by all pairs of practitioners.



It should be mentioned that the best performing pair (SW-P 1 and 5) had previous practical experience with conducting usability evaluations, see Table 1**.**
By removing all pairs consisting of SW-P 1 or 5 we see that the average number of identified problems is lowered to 33.3 (SD=4), which amounts to 66.7 % of all problems. Considering the severity categorizations we find that the average number of critical problems identified for all SW-P pairs is 9.2 (SD=1.9), 14.8 (SD=2.4) for serious problems and 11.7 (SD=3.1) for cosmetic problems, or 77 %, 78 % and 62 % respectively.

4.3 Quality of problem descriptions

This subsection describes the software development practitioners' ability to describe usability problems according to the five quality attributes of clarity, impact, data support, cause and user actions, which are derived in [3].

Table 2 provides an overview of the median quality ratings given by the three external raters where higher ratings indicate a higher level of fulfillment according to the quality attributes (1-5 scale). The table shows that problem descriptions written by practitioners 1 and 5, who received the median scores of 4 and 3 respectively, described their usability problems with a quality comparable to that of the HCI specialist (median = 4). The other three practitioners scored lower as their median rating was 2. The table also shows that practitioners are better at being clear and precise (clarity) in their problem lists than any of the other attributes, which is elaborated upon below along with qualitative comments made by the external raters.

Table 2. Median quality ratings given by the three external raters to the problem lists written by the software development practitioners (SW-P) and the HCI specialist.

Participant	Clarity	Impact	Data	Cause	Actions	Overall median
SW-P 1	4	3	4	4	4	**4**
SW-P 2	2	2	2	2	2	**2**
SW-P 3	2	2	2	2	1	**2**
SW-P 4	3	2	2	2	2	**2**
SW-P 5	4	2	3	3	3	**3**
Overall median	**3**	**2**	**2**	**2**	**2**	**2**
HCI specialist	**4**	**3**	**4**	**3**	**5**	**4**

Clarity. Table 2 shows that the practitioners were better at fulfilling the clarity attribute than any of the other attributes as they scored an overall median of 3. In comparison the HCI specialist received the median rating of 4 by the external raters. This was also the case for practitioners 1 and 5. As an example on the qualitative comments given, one of the raters mentioned that 5's list provided *"Good insights in the problems experienced"*. Practitioners 2, 3 and 4 scored the lowest median ratings on this attribute where one rater mentioned the following about practitioners 3's list: *"Extremely short and imprecise descriptions. Actually the descriptions are so poor that you in most cases cannot find out what the problem is"*.

Impact. Table 2 also shows that lower median ratings were given with respect to the impact attribute compared to clarity, which is the case for both the practitioner and HCI specialist descriptions. Practitioners got an overall median of 2 and the HCI specialist 3. Practitioner 1 performed on par with the HCI specialist on this matter and got a higher median rating than the remaining four. One of the external raters commented that practitioners in some problems describe the impact on the user's task but other elements such as business effects and affected system components are left unmentioned. This is also the case for descriptions provided by the HCI specialist.

Data Support. Practitioners' descriptions received an overall median rating of 2 by the external raters where practitioner 1 and 5 scored highest (4 and 3 respectively). In comparison the HCI specialist received the median rating 4 on this quality attribute. One of the raters commented that practitioners in general describe how many test users that experience given problems and that they in certain descriptions state whether or not the task was a success or a failure. Another mentioned that: *"Many problems are not clearly connected to observations"*, thus this rater found that practitioners did not always consider objective data. The same rater additionally mentioned that practitioners made use of vague statements such as: *"The user does not understand"* or *"the user is in doubt"*, statements which are of a speculative nature. However, the practitioners did describe how many test users that experienced the problems and whether or not the tasks were completed, which is similar to the information provided by the HCI specialist. Additionally it was commented that the HCI specialist provided *"good descriptions of the critical incidents"*.

Problem Cause. On this attribute an overall median rating of 2 was given on practitioners' descriptions and the HCI specialist received a median of 3. Practitioners 1 and 5 once more scored higher median ratings than the other three. One of the external raters mentioned the following about practitioner 1's descriptions: *"The list is ok with good descriptions that to a great extent describe causes"*, which was agreed upon by another rater. The third rater, however, found that this practitioner was guessing on the users' thoughts and the cause of the problem in some of his descriptions. Practitioners 2, 3 and 4 were given the lowest ratings in which case all three raters agree that no causes or arguments are provided.

User Actions. Finally Table 2 shows that practitioners and the HCI specialist received median ratings of 2 and 5 respectively on this attribute. Two of the raters mentioned that several of the practitioner descriptions provided examples on users' navigational flow, but that reactions are sometimes described implicitly by stating that users *"are in doubt"* or *"overlooks"* certain elements in the interface. However, according to one of the raters, practitioners 2 and 3 do not describe user reactions at all. Yet again practitioners 1 and 5 scored the highest ratings compared to the other practitioners, where they received medians of 4 and 3 respectively. Two raters found that the descriptions written by the HCI specialist contained detailed information on users' navigational flow and reactions.

5 Discussion

Findings from this study suggest that practitioners are able to identify 48.4 % of all usability problems where the one who identified most problems found 66 % and the one who identified fewest found 28 %. Considering related work, the studies presented in [1], [7] and [21] show that university students are able to identify between 11 % and 33 % of all problems. We additionally found that practitioners on average discovered 78 % of the problems identified by the HCI specialist. In comparison study presented in [20] show that students identified a mean of 37 % of the problems identified by specialists. Thus, in our study we see that the performance of software development practitioners performed closer to the HCI specialist compared to findings in related work.

As mentioned previously, it can be too resource demanding in practice to utilize five evaluators in analysis of usability data, which is why we also examined how many problems each pair of practitioners identified. Our study shows that the most effective pair found 86 % and the least effective found 52 %, where the average was 71.4 %. Also, looking at the number of problems in each severity category, we found that, on average, all pairs identified 77 % of the critical problems, 78 % of the serious and 62 % of the cosmetic. In comparison the HCI specialist identified 50 %, 63 % and 68 % of the critical, serious and cosmetic problems respectively. Thus, we see that two software development practitioners from this study are able to identify more critical and serious problems than the HCI specialist while they have comparable performance with respect to cosmetic. To validate the performance of the HCI specialist in our experiment we found a study conducted by Jacobsen and colleagues which shows that four specialists conducting video based analysis identified an average of 52 % of all problems [10]. This is comparable to the 62 % identified by the specialist in our study. In relation to this it should be mentioned that SW-P 1 and SW-P 5 was the pair that identified most problems (86 %), a finding which may be explained by the fact that they had practical usability experience from their education (7 and 13 years ago respectively, see Table 1). Thus, it could be argued that these practitioners are not novices compared to the participants applied in related work. However, our results indicate, that even by removing all pairs consisting of SW-P 1 or 5 we still find that a pair of practitioners on average perform better than the HCI specialist in terms of number of identified problems.

In the above we have compared the performance of the software development practitioners in this study to that of students', which are used as the empirical basis in related work. The higher level of thoroughness of the practitioners in our study could be caused by differences in the amount of training given and in [7] the students received 6 – 9 hours of training in the form of reading instructions of the methods to be applied. The 14 hours given in our two-day course as a combination of theory and exercises differs considerably from this. On the other hand it is reported in [20] that students received 40 hours of training as a combination of lectures and exercises. Another cause for the differences may be motivational factors, as software development practitioners, due to a competitive market, are more dependent on increased sales of their software products than university students. Also, students may lack incentive in cases where they do not receive payment or if the experiment is part of a mandatory course, a notion which is supported in [20].

Findings also revealed that practitioners on average were unable to fulfill the quality attributes in their problem descriptions to the same degree as the HCI specialist. Exceptions to this, however, were SW-P 1 and 5 who provided a quality comparable to the specialist, which as mentioned earlier may be caused by their previous experience with usability evaluations. Still, the average result corresponds to the findings in [20] in which it is reported that qualitative aspects of the problem descriptions written by students are poorer than that of HCI specialists. Our study extends this quality assessment by dividing it into the five quality attributes mentioned in [3]. This enables us to express that practitioners were better at providing clear and precise problem descriptions than they were at describing the impact, cause, user actions and providing data support for observations. A reason for this may be located in the fact that some of the software development practitioners in our study are used to provide code comments in their software. During one of the debriefing interviews a practitioner mentioned: *"I find it important to write understandable comments because it's easier to get back into the code if you've had one or two weeks of vacation"*. Thus, clarity as a quality attribute is important to industry practitioners in a different context which could indicate why they fulfill the clarity attribute better than any of the other.

6 Conclusion

This exploratory study indicates that cross pollinating usability and software development disciplines may be accomplished by training software development practitioners. Findings show that the practitioners after a two-day training course gained key knowledge on how to conduct usability evaluations as they were able to identify a mean of 48.4 % of all usability problems and that two practitioners are able to identify 71.4 %. This exceeded the performance of an HCI specialist, who identified 62 % of all problems. We also observed that practitioners were better at providing clear and precise problem descriptions than they were at describing the impact, cause, user actions and providing data support for observations. Their problem descriptions, however, were of a lower quality compared to the specialist with the exception of two practitioners.

Findings from this study should be backed up by further studies based on more participants. Also, as our study is conducted at a fixed point in time, we still need studies of long term effects of letting such practitioners do the testing in order to validate that such cross pollination will be carried out in everyday work situations. Also, it would be interesting to conduct further studies on learning retention, e.g. how knowledge within the area increases or diminishes over time.

References

1. Ardito, C., Costabile, M.F., De Angeli, A., Lanzilotti, R.: Systematic evaluation of e-learning systems: an experimental validation. In Proc. NordiCHI 2006, pp. 195-202. ACM Press, New York (2006).

2. Bruun, A.: Training Software Developers in Usability Engineering: A Literature Review. In Proc. NordiCHI 2010, pp. 82-91. ACM Press, New York, NY, USA.
3. Capra, M.G.: Usability Problem Description and the Evaluator Effect in Usability Testing. Virginia Polytechnic Institute & State University, Blacksburg (2006).
4. Edwards, A., Wright, P., Petrie, H.: HCI education: We are failing–why? In Proc. HCIEd, pp. 127-129. Springer (2009).
5. Fonseca, M., Jorge, J., Gomes, M., Gonçalves, D., Vala, M.: Conceptual design and prototyping to explore creativity. In IFIP, vol. 289, pp. 203-217. Springer (2009).
6. Frøkjær, E., Hornbæk, K.: Metaphors of human thinking for usability inspection and design. In TOCHI, vol. 14, issues 4. ACM Press, New York (2008).
7. Frøkjær, E., Lárusdòttir, M.K.: Prediction of Usability: Comparing Method Combinations. In Proc. IRMA. Idea Group Publishing, Hershey (1999).
8. Gulliksen, J., Boivie, I., Persson, J., Hektor, A., Herulf, L.: Making a difference: a survey of the usability profession in Sweden. In Proc. NordiCHI 2004, pp. 207-215. ACM Press, New York (2004).
9. Häkli, A.: Introducing user-centered design in a small-size software development organization. Helsinki University of Technology, Helsinki (2005).
10. Jacobsen, N.E., Hertzum, M., John, B.E.: The Evaluator Effect in Usability Studies: Problem Detection and Severity Judgments. In Proc. of HFES, pp. 1336-1340. HFES, Santa Monica (1998).
11. Høegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B., Stage, J.: The Impact of Usability Reports and User Test Observations on Developers' Understanding of Usability Data: An Exploratory Study. In Int. Journal of Human-Computer Interaction, vol. 21, issues 2, pp. 173-196. Taylor & Francis (2006).
12. Ji, Y. G., Yun, M. H.: Enhancing the minority discipline in the IT industry: A survey of usability and User-Centered design practice. In Int. Journal of Human-Computer Interaction, vol. 20, issue 2, 117-134. Taylor & Francis (2006).
13. Juristo, N., Moreno, AM, Sanchez-Segura, M. I.: Guidelines for eliciting usability functionalities. In IEEE Transactions on Software Engineering, vol. 33, issues 11, pp. 744-758. IEEE Computer Society Press (2007).
14. Metzker, E., Offergeld, M.: An Interdisciplinary Approach for Successfully Integrating Human-Centered Design Methods Into Development Processes Practiced by Industrial Software Development Organizations. In Proc. IFIP International Conference on Engineering for Human-Computer Interaction, pp. 19-34. Springer-Verlag, London (2001).
15. Molich, R. Usable Web Design. Nyt Teknisk Forlag (2007).
16. Nielsen, J.: Finding usability problems through heuristic evaluation. In Proc. CHI, pp. 373-380. ACM Press, New York (1992).
17. Rosenbaum, S., Rohn, J. A., Humburg, J.: A toolkit for strategic usability: results from workshops, panels, and surveys. In Proc. CHI 2000, pp. 337-344. ACM Press, New York (2000).
18. Rubin, J., Chisnell, D.: Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests, 2nd. Edition. John Wiley & Sons, Inc., Indianapolis (2008).
19. Seffah, A., Gulliksen, J., Desmarais, M.: An Introduction to Human-Centered Software Engineering. In Seffah, A., Gulliksen, J., Desmarais, M.C. (eds.), Human-Centered Software Engineering — Integrating Usability in the Software Development Lifecycle, Human-Computer Interaction Series, Vol. 8, pp. 3-14. Springer, Netherlands (2005).
20. Skov, M. B. and Stage, J., 2008. Direct Integration: Training Software Developers and Designers to Conduct Usability Evaluations. In Proc. of the First Workshop on the Interplay between Usability Evaluation and Software Development. CEUR-WS.org.
21. Wright, P.C., Monk, A.F.: The Use of Think-Aloud Evaluation Methods in Design. In ACM SIGCHI Bulletin archive, vol. 23, issue 1, pp. 55 - 57. ACM Press, New York (1991).

Training Software Development Practitioners in Usability Testing: An Assessment Acceptance and Prioritization

Anders Bruun
Dept. of Computer Science, Aalborg University
Selma Lagerlöfs Vej 300, DK-9220 Aalborg
bruun@cs.aau.dk

Jan Stage
Dept. of Computer Science, Aalborg University
Selma Lagerlöfs Vej 300, DK-9220 Aalborg
jans@cs.aau.dk

ABSTRACT
Previous studies show that some software development practitioners without a usability background experience difficulties in understanding users and accepting that usability problems exist in their software. Also they do not prioritize fixing the problems identified by specialists. This barrier is denoted "developer mindset" and this paper empirically explores whether the barrier can be overcome by training software development practitioners to conduct usability testing. Findings show that the practitioners obtained considerable abilities in identifying usability problems as they managed to uncover 80.5 % of these after 30 hours of training, which shows a high level of acceptance. Findings also reveal that the practitioners prioritized fixing 61 % of the problems and we found that they successfully removed 64 % in a new interface design. We conclude that this approach may pose a viable solution to overcome the barrier of developer mindset.

Author Keywords
Usability Testing, Training, Software Development Practitioners, Developer Mindset

ACM Classification Keywords

H.5.2 [Information Interfaces and Presentation]: User Interfaces – Evaluation/methodology, Theory and methods.

INTRODUCTION
The level of commitment devoted to fix identified usability problems is important in practice as it is one of the main parts in determining whether a usability test has been a success or not (Wixon 2003). Previous studies have adopted the concept of "downstream utility" to determine the extent to which results from usability tests impacts the usability of a system, cf. (Law 2006; Sawyer et al. 1996).

A recent survey conducted in 39 Danish software companies identified considerable barriers to introducing usability testing within their development processes (Bak et al. 2008). The most prominent of these barriers is denoted "developer mindset". This describes the tendency that software development practitioners without a usability background are having difficulties in understanding users, accepting results from usability tests and prioritizing fixing problems (Bak et al. 2008). The barrier of developer mindset is also found in an Italian study equivalent to the Danish one (Ardito et al. 2011). This barrier compromises the commitment devoted to fix identified usability problems, which in turn leads to a low downstream utility.

A possible solution to overcome this barrier may be to involve software development practitioners in the execution of usability tests. A study conducted by Høegh and colleagues supports this by showing that developers' awareness of usability problems was increased after they observed sessions of user tests (Høegh et al. 2006). A reason for the increased awareness was that the observations of user tests provided firsthand experiences of how the system was applied in the real world by real users (Høegh et al. 2006). Involvement can also go beyond observation, e.g. by letting such software development practitioners actively participate in the testing process (Bruun & Stage 2011; Häkli 2005). Previous studies have shown promising results on letting such practitioners participate actively in usability activities. Metzker and Offergeld describe findings from a case study where software development practitioners together with usability specialists performed contextual task analysis. This motivated the practitioners to emphasize usability aspects (Metzker & Offergeld 2001). The study presented in (Bruun & Stage 2011) takes this a step further by presenting a study where similar practitioners were trained to conduct user based tests without the presence of usability specialist.

In this paper we present an exploratory study on training software development practitioners to conduct user based usability tests. The paper extends previous research by examining whether this approach could be a viable solution to overcome the barrier of developer mindset. In particular we aim to measure how the approach affects downstream utility.

The paper is structured as follows. First an overview of related work on training students and industry practitioners in user based usability testing methods is provided. This is followed by a description of the experimental method applied after which findings are presented and discussed in relation to related work. Finally we provide the conclusions and point out avenues of future work.

OZCHI'12, November 26–30, 2012, Melbourne, Victoria, Australia.

RELATED WORK

This section provides an overview of empirically based related work in which participants received training to conduct user based usability tests. This is divided in three parts based on the research focus in the papers. The first describes studies of tools, the second presents studies of methods and the third describes studies of training.

Studies of Tools

Three papers present studies of either software tools or conceptual tools that assist evaluators in identifying usability problems. Two of them are based on the same experiment and describe the development and evaluation of a software tool to ease the transformation of raw usability data into usability problem descriptions (Howarth 2007; Howarth et al. 2007). In the experiment 16 graduate students were selected to act as evaluators and were evenly distributed to use either of two tools to note problems. They received one hour training in order to familiarize themselves with the tools after which they were asked to view videos from an earlier usability test. Findings from these studies relate to the quality of problems problem descriptions and show that students were better at formulating of user actions than by providing clarity, data support etc. in their problem descriptions.

The third paper describes development and evaluation of a conceptual tool to support problem identification is presented in (Skov & Stage 2005). This tool is not software based and is mentioned as being a conceptual tool and is represented by a 4x3 matrix that evaluators can use when observing users. A comparative study was conducted using 14 undergraduate students who were distributed over two experimental conditions, one in which the conceptual tool was applied to test a user interface and another without the tool. Students then conducted a usability test based on recordings of a user applying a web-based system to solve a series of tasks. Results from that study show that students were able to identify 18 % of all problems and that they discovered a mean of 20 % of the problems identified by two usability specialists.

Studies of Methods

Three papers present comparative studies of usability testing methods. Koustabasis and colleagues describe an experiment in which the focus is on evaluating the performance of students with respect to number of identified problems, validity and efficiency (Koustabasis et al. 2007). Four different methods were compared: Heuristic Inspection, Cognitive Walkthrough, user based test and Co-discovery learning. Twenty-seven students were distributed over the four conditions. That study indicated that students applying the user based method were able to identify 24 % of all problems on average (Koustabasis et al. 2007).

The second comparative study of usability testing methods is presented in (Ardito et al. 2006) and describes the development and evaluation of the eLSE method for testing a system for e-learning. Seventy-three senior students were used as the empirical basis for comparing the performance of the following methods: eLSE, User Based Test and Heuristic Inspection. Findings show that students applying the user based method identified an average of 11 % of all problems (Ardito et al. 2006).

The third paper describing a comparative study of usability testing methods is (Frøkjær & Lárusdòttir 1999). The focus of this paper is on comparing the effect of combining methods. Participants were in one condition asked to apply Cognitive Walkthrough followed by a second round applying a user based test. In a second condition other participants applied Heuristic Inspection followed by user based testing. Fifty-one students formed the empirical basis and results indicate that students were able to identify 18 % of all problems.

Studies of Training

The focus within five papers is on training participants to analyze data from user based tests. Two of these papers are based on the same experiment in which 36 teams of first year students were trained in how to conduct usability tests (Skov & Stage 2004; Skov and Stage 2008). The students had received 40 hours of training before participating in the experiment and they were instructed to conduct a test, analyze the results and to write a report documenting all steps in the process. Student reports were compared to reports written by 8 usability expert teams. Results from these two studies indicate that students identified a mean of 7.9 problems and that specialists found a mean of 21. Thus, on average students found 37 % of the problems identified by specialists.

The study presented in (Wright & Monk 1991) describes two experiments with the purpose of examining the application of user based tests. The first experiment concerns how effective usability testing is when applied by software trainees after reading a short manual. Trainees documented their results in a report which was then assessed with respect to the number of identified problems and their severity. Performance was then compared to that of usability specialists'. The second experiment in that paper examines the differences of testing own design versus the design made by others. Two new groups of trainees were divided in two conditions and in the first, participants were asked to design and test their own prototype and in the second they tested the designs made by participants in the first condition. Like the first experiment, reports were assessed with respect to the number of identified problems. Findings showed that each student team identified 33 % of all problems on average (Wright & Monk 1991).

The study presented in (Bruun & Stage 2011) examines the ability of software development practitioners from industry in identifying and describing usability problems. In that study 5 practitioners received a two day training course on user based tests and how to conduct video analysis. Findings show that practitioners on average were able to uncover 48.4 % of all usability problems. It also showed that two practitioners on average could identify 71.4 %, which exceeded the performance of a single usability specialist, who identified 62 %.

The final study is a master thesis which had the objective of introducing a user-centered method into a small software company and to increase the knowledge of software developers on the matter (Häkli 2005). The researcher did in this case conduct a 14 hour training course for 13 software development practitioners on interaction design, prototyping, Heuristic Inspection and user based testing. Emphasis in this study is put on participants' performance of conducting Heuristic Inspection.

Summarizing on the related work it can be seen that the main focus of these papers regard university students' performance in conducting usability tests. Exceptions to this are (Bruun & Stage 2011) and (Häkli 2005) which use software development practitioners as the empirical basis for evaluating novice performance, thus, there is a need for further studies on software development practitioners' ability to identify usability problems. We also find a need for studying how such an approach affects developer mindset in terms of acceptance and prioritization of usability problems.

METHOD

This section describes the experimental procedure followed within this study. The overall idea was to train the software development practitioners (henceforth also mentioned as "practitioners") to conduct user based usability tests and then evaluate two versions of the same system. They tested the first version of the system after which they spent 3 months fixing the problems identified. The time span of 3 months was selected so that the practitioners had sufficient time to fix the problems. After 3 months a second usability test was conducted to determine the effectiveness of the fixes.

Training Courses

Basic training course

The practitioners participating in our study had no practical experience in conducting usability tests (further details are provided the participants section below). For that reason we focused on teaching a traditional user based test with video analysis as described in (Rubin and Chisnell 2008). This two-day (14 hours) training course was held by the authors of this paper as a combination of presentations and exercises. To conclude the basic training course the practitioners were asked to do a homework assignment of analyzing five video clips from a previous usability test. The practitioners spent an average of 8.8 hours on this task. The resulting problem lists were reviewed and we provided feedback to the practitioners on how to improve their problem descriptions.

Follow-up training course

The traditional usability test method taught during basic training necessitates traversing several hours of video data, which require a considerable amount of resources. For this reason we also chose to train the practitioners in applying Instant Data Analysis (IDA), as this method is not based on reviewing video data. IDA is conducted at the end of a test and involves the following steps, cf. (Kjeldskov et al. 2004):

- *Brainstorm:* The test monitor and data loggers participating in the test identifies the usability problems they can remember while one of them notes problems on a whiteboard.
- *Task review:* The test monitor and data loggers review all tasks to recall additional problems that occurred.
- *Note review:* The data loggers review their notes to remember further problems.
- *Severity rating:* The test monitor and data loggers discuss the severity of the problems and rate these as critical, serious or cosmetic, cf. (Molich 2000).

This one-day (7 hours) follow-up course in IDA was held by the authors two months after the basic training. A combination of presentations and exercises was also applied in this course.

Participants

Software development practitioners

Three software development practitioners from a small Danish software company (< 25 employees) participated in this study. This company had previously taken part in the survey presented in (Bak et al. 2008) which showed the existence of developer mindset. Two of the practitioners worked as project managers but also had responsibilities as software developers while one worked as a software developer exclusively. Two had no previous knowledge about usability tests while one had theoretical knowledge from an HCI course taken during his education. Thus, none of them had previous practical experience in conducting usability tests.

Test users

A total of seven test users participated in the two tests, all of which were recruited by the practitioners. Three test users participated in the first test and four in the second. The test users were employed as administrative staff within different companies and all had experience in applying for wage subsidies, a process which is supported by the evaluated system (the system is further elaborated in the system section below). None of them had used the system before.

Usability specialists

Three usability specialists analyzed the video material obtained from both tests in order to evaluate the performance of the practitioners. Two of these were external usability specialists employed in industry and the third was an HCI researcher (one of the authors). The external usability specialists had not otherwise taken part in the experiment. None of the specialists had previous experience from the domain of wage subsidies.

Conduction of the Usability Tests

This section describes how the two usability tests were conducted by the three software development practitioners. As mentioned previously, these tests were conducted 3 months apart.

Planning
The three practitioners planned and conducted the two tests including finding the test users as well as defining the 3 tasks given to these. The same 3 tasks were given in both tests. They also distributed the roles between them, i.e. who would act as test monitor and who would be the data loggers noting down problems during the tests.

System
The system evaluated was a web application used by administrative staff within companies to register and apply for wage subsidies. Applications filled out by the administrative staff are then submitted to the local municipality, which then provides companies with a subsidy for the employees enrolled in such a settlement. The system consisted of two parts: 1) A stepwise wizard in which the data would be entered and 2) a pdf form shown as a confirmation at the end of the wizard, in which users could edit previously entered data. The system was developed by the small software company in which the practitioners were employed.

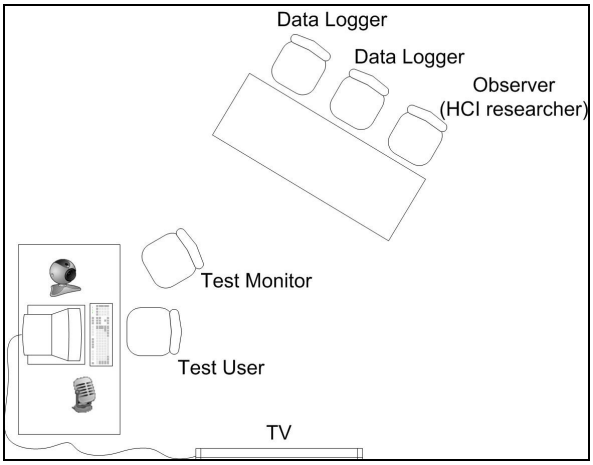


Figure 1. Overview of test settings.

Setting
Figure 1 shows an overview of the setting applied in the tests, which were conducted in an office at the company. Video capture software was utilized to record user interaction within the system and a webcam recorded the face of the test users. An external microphone was applied to capture audio. During each session a test user was sitting at a table in the office solving the predefined tasks by using the system. One practitioner acted as test monitor and sat next to the test user. The two others acted as data loggers by noting down usability problems and observed the interaction through a projection on a 50" TV screen within the office. The data loggers along with one of the authors who observed the sessions sat 4 meters behind the test user in order not to interfere.

Procedure
Each of the two tests was conducted in one day. For each test session a user would firstly get introduced by the test monitor to the test procedure and the system. Then the user was asked to solve the 3 tasks one while thinking aloud. If the user did not think aloud, the test monitor prompted her/him to do so. Each test session lasted 30 – 45 minutes.

At the end of each test day the practitioners conducted Instant Data Analysis to derive the lists of usability problems. During both analysis sessions, one of the data loggers acted as facilitator by noting and organizing the identified usability problems on a whiteboard.

Improving the System
The list of usability problems from the first test was used by the practitioners as input to improve the usability of the system. Two days after the first test they held a one day meeting in which they discussed which problems to address and how to redesign the system. This was followed by three months of development, which was mainly done by the practitioner without project management responsibilities. All practitioners held weekly meetings during the period of development.

Interviews
After the second test the three practitioners were interviewed by one of the authors. The interview was semi structured and was conducted with all practitioners present at once in order to foster discussions. The purpose was to uncover the factors influencing the practitioners' prioritization of fixing the problems and reasons for why some problems recurred in the second test. Audio recordings from the interview were transcribed by one of the authors and analysed using grounded theory at open coding level (Strauss & Corbin, 1998).

Analysis of Problem Lists
The three usability specialists (one of the authors and two external) analyzed the seven video recordings from the two tests. Videos were analyzed in random order to reduce ordering bias. The specialists applied the same document template as the practitioners for describing problems. The severity of each problem was categorized as either cosmetic, serious or critical, corresponding to the categorizations applied by the practitioners. Analysis was firstly done individually where each specialist created two problem lists; one for the first version of the system and another for the second version. After the individual analysis all specialists held a meeting in which their lists were merged into two lists consisting of all identified problems (one total list for the first version of the system and a similar list for the second version). During the merging process the specialists negotiated the severity of the problems until agreement was reached. Finally the two lists created by the practitioners through the IDA sessions were merged into the lists created by the specialists. In case of identical problems, the severity ratings given by the practitioners were overridden by that given by the specialists. Severity ratings on problems identified by the practitioners only were not changed.

Measuring Developer Mindset
Developer mindset describes the tendency that some practitioners without a usability background are having difficulties in accepting results from usability tests as well as prioritizing these. This section describes the

measurements utilized in our analysis to determine the level of developer mindset after training the practitioners. In our analysis we find the concept of "downstream utility" suitable for determining acceptance and prioritization. The downstream utility of a test can be expressed in terms of the Committed Impact Ratio (CIR) and Completed-to-Date Impact Ratio (CDIR), cf. (Law 2006; Sawyer et al. 1996). CIR denotes the extent to which a development team commits to fixing usability problems before an actual implementation takes place and is calculated as follows:

$$CIR = \frac{No.\ of\ problems\ comitted\ to\ fix}{Total\ no.\ of\ problems\ found} \cdot 100$$

CDIR is a measure of the usability problems actually fixed at a given point, i.e. after implementation has begun:

$$CDIR = \frac{No.\ of\ problems\ fixed}{Total\ no.\ of\ problems\ found} \cdot 100$$

RESULTS

This section presents our findings in relation to practitioner thoroughness in identifying usability problems as well as the downstream utility.

Thoroughness

Table 1 and Table 2 show an overview of the number of problems identified by the practitioners and specialists in the two tests. In total the practitioners and specialists identified 41 problems in the first version of the system (test 1) and 44 in the second (test 2). The practitioners identified 81 % and 80 % of all problems in test 1 and 2 respectively while the specialists identified 76 % and 73 %. A Fisher's Exact test reveals no significant differences between the thoroughness of the practitioners and specialists in the first test (df=1, p>0.7). This is also the case for the second test (df=1, p>0.6).

Test 1

	Critical		**Serious**		**Cosmetic**		**Total**	
	#	*%*	*#*	*%*	*#*	*%*	**#**	**%**
Prac.	8	73	13	87	12	80	**33**	**81**
Spec.	10	91	14	93	7	47	**31**	**76**
Total	**11**	**100**	**15**	**100**	**15**	**100**	**41**	**100**

Table 1: Number of problems identified by practitioners and specialists in the first test.

In both tests the practitioners identified fewer critical and more cosmetic problems than the specialists. In terms of serious problems, the practitioners identified one less in the first test and two more in the second. In addition, findings reveal that the number of critical and serious problems was almost halved in the second version of the system as the count decreased from 26 to 15. However, the number of cosmetic problems was doubled as they increased from 15 to 29.

Test 2

	Critical		**Serious**		**Cosmetic**		**Total**	
	#	*%*	*#*	*%*	*#*	*%*	**#**	**%**
Prac.	5	63	7	100	23	79	**35**	**80**
Spec.	7	88	5	71	20	69	**32**	**73**
Total	**8**	**100**	**7**	**100**	**29**	**100**	**44**	**100**

Table 2: Number of problems identified by practitioners and specialists in the second test.

Table 3 shows an overview of the thoroughness of each of the three usability specialists. On average the specialists identified 48 % and 41.7 % of all problems found in test 1 and 2 respectively. Due to the plenary nature of the IDA session, such an overview cannot be made for the practitioners.

	Test 1		**Test 2**	
	# (SD)	*% (SD)*	*# (SD)*	*% (SD)*
Specialist 1	26	63	25	57
Specialist 2	16	39	14	32
Specialist 3	17	42	16	36
Average	19.7 (5.5)	48 (13.1)	18.3 (5.9)	41.7 (13.4)

Table 3: Number of problems identified by the usability specialists. Percentages are of the total no. of problems found in each test, (SD) = Standard Deviation.

Problem Agreement

In the first test 23 of the 41 problems were identified by the practitioners and the specialists, hereby leading to problem agreement of 56 %. In the second test the two groups also had 23 of the 44 problems in common, which gives an agreement of 52 %. Thus, the average agreement is 54 % (SD=2.8).

Differences in Severity Categorizations

As mentioned above the practitioners and specialists had an agreement on 23 problems in the first test. The severity ratings (critical, serious or cosmetic) given by the two groups differed in 16 (70 %) of these problems where the practitioners consistently gave lower ratings than the specialists. In the second test there was a disagreement on severity ratings in 5 of the 23 problems (22 %) where the practitioners once more provided lower ratings than the specialists.

Uniquely Identified Problems

Across the two tests 22 problems are identified by the practitioners only, of which 2 are critical, 3 serious and 17 cosmetic. In the following we provide an example of one of the serious problems. Two types of information are needed in the system in order to apply for wage subsidies. The first is related to the base salary, which includes the subsidy given by the municipality plus the amount paid by the employer while the other relate to the amount given by the employer only. During the tests the

practitioners noted that some test users did only use the first type of information, which is not enough to submit a correct application form. The above mentioned example is highly domain specific and requires additional knowledge in order to be uncovered, especially since the users did not notice the problem themselves and, hence, did not comment on this explicitly during the test. Other similar problems were identified by the practitioners but not by the usability specialists.

Committed Impact Ratio

The problem list made by the specialists was not available before starting the implementation, which meant that the practitioners only had access to their own list of problems derived through the IDA session from the first test. For this reason we in the following base the total number of identified problems on the 33 problems found by the practitioners, see Table 1. Before starting the implementation the practitioners committed to fix 20 problems, which results in the following committed impact ratio:

$$\text{CIR} = \frac{20}{33} \cdot 100 = \mathbf{61\ \%}$$

In the interview conducted at the end of the experiment we asked the practitioners of what factors had influenced the CIR, i.e. their prioritization of fixing usability problems. These factors were derived from existing literature and regarded: Severity ratings, frequency, length of problem descriptions and resource requirements, cf. [**Fejl! Henvisningskilde ikke fundet.**].

The interviews revealed that the amount of resources required to fix a usability problem was one of the main factors in prioritizing these. One of the practitioners for instance mentioned that: *"... it didn't matter what severity rating the problems had but if it was a problem that could easily be corrected, it would come on the list of fixes"*. Another practitioner followed up on this by saying: *"Yes, and in the opposite case we have the problems which could cause great technical challenges. Those problems are on stand-by, not forgotten, but put into the log for future corrections"*. Based on this we see that the factor of resource requirements had higher influence than severity ratings.

Frequency in terms of the number of users experiencing a problem was not an influential factor, a practitioner for instance mentioned: *"If we have had ten test users and a problem was experienced by one of these it would be a different assessment compared to the three or four users we had... In our case we chose to say that if one user experiences the problem, it can also happen for others"*. Thus, problems experienced by three test users would not be emphasized over those experienced by a single test user.

The practitioners also mentioned that frequency, measured as the number of problems found within a given system component, influenced their priorities, e.g.: *"After our first test we saw a lot of problems concerning the dates... The calendar component. It was that component that we spent the most time on improving"*.

The practitioners stated that the length of the problem descriptions did not influence their prioritization. As an example, one stated that: *"In the analysis we did not have problems where we said 'what was this problem?'"*, to which another replied: *"Yes, and the analysis is conducted immediately after the sessions so we do remember them"*.

An additional factor identified regarded the coherence to other systems in the company portfolio. The company was developing a new platform on which to base existing solutions, and if a usability problem was deeply rooted in the design of this old platform, it would not be prioritized, e.g.: *"... you can also say that what has happened in some cases was that we said 'but we will not fix this now as the new framework will come out later'"*. In Relation to this the practitioners only prioritized fixing problems related to the part of the system containing the stepwise wizard and not the editable pdf form.

In sum, the practitioners in our case mainly committed to fixing problems based on the factors of resource requirements and coherence to other systems while it did not matter whether a problem was experienced by a single or multiple test users. Finally, severity ratings and length of problem descriptions were less influential.

Completed-to-Date Impact Ratio

We found that 12 out of the total of 33 problems identified by the practitioners during the first test recurred in the second version. Thus, 21 problems are fixed, which gives the following completed-to-date impact ratio:

$$\text{CIR} = \frac{21}{33} \cdot 100 = \mathbf{64\ \%}$$

During the interview we asked the practitioners why 12 problems from the first version of the system recurred in the second. One of the reasons was that 4 of these problems were related to the editable pdf form, which, as mentioned above, was not prioritized.

In case of the other 8 recurring problems, the practitioners mentioned they tried to fix 5 of these, but that the implemented fixes did not work as intended. One of the problems was related to the help texts, which lacked necessary information to which they mentioned: *"We have tried to make these more elaborate... We went through all of the texts to see if they properly explained the wordings"*.

The interviews also revealed that one of the recurring problems was not accepted by the practitioners as one of them mentioned: *"Well you could say that we should have taken this problem more seriously after the first test, so we should have dug deeper into this already at the first test, just like we did after we found it again"*.

The final two recurring problems were not fixed as possible solutions conflicted with the usability of other system components or features prioritized by the sales department. As an example, one of these relates to the introduction presented in the system, which was not read by the test users due to its length. The solution of reducing the amount of text was not followed as this

conflicted with the usability of another system component, one mentioned: *"With the introduction we also try to solve another problem about attachments. The introduction should avoid the users from getting stuck in the middle of the wizard because we let them know up front what attachments they need"*.

Summarizing on the above we found that the practitioners have tried to fix most of the prioritized problems that recurred, but that these fixes did not work as intended. Additionally, one of the problems was not accepted after occurring in the first test, but was then prioritized after its presence in the second.

DISCUSSION

Findings of our study show that the practitioners after receiving 30 hours of training were able to identify an average of 80.5 % of all usability problems across the two tests. In general, related work report of a lower thoroughness than the one found in our experiment. The study presented in (Bruun and Stage 2011) showed that 5 practitioners from industry on average were able to uncover 48.4 % of all usability problems and that two practitioners on average identified 71.4 %. In (Wright and Monk 1991) it is shown that each student team identified 33 % of all problems on average. In the study conducted by Koustabasis and colleagues it was found that students applying the user based method were able to identify 24 % of all problems on average (Koutsabasis et al. 2007). In (Frøkjær and Lárusdòttir 1999) it is shown that students were able to identify 18 % of all problems whereas the level of thoroughness reported in (Ardito et al. 2006) is lower as the students that applied a user based method identified an average of 11 %. The two studies presented in (Skov and Stage 2004; Skov and Stage 2008) compare student performance to that of specialists and show that students identified a mean of 37 % of the problems identified by specialists.

We see that the practitioners in our experiment were able to identify more problems than in the above mentioned studies. Differing motivational factors could play a considerable role in the variations between related work and our study. In a competitive market, software development practitioners are highly dependent on sales of their products, which is not the case for university students, which, with the exception of (Bruun and Stage 2011), are used in the above mentioned related work.

We also found that the three practitioners had a slightly higher thoroughness than the three specialists. A reason for this could be the thoroughness of the usability specialists within our study where each on average revealed 45 % of all problems. This, however, is comparable to the thoroughness presented in (Jacobsen et al. 1998) where four specialists conducting video based analysis identified an average of 52 % of all problems.

Another explanation of why the practitioners managed to identify more problems than the specialists could be that they had a higher level of domain knowledge. Our findings show that 22 problems were uniquely identified by the three practitioners and that some of those problems related to domain specific issues. This indicates one of the benefits of letting software development practitioners conduct usability tests compared to externally hired specialists, as their domain knowledge contributes in uncovering additional problems. This is supported in (Følstad and Hornbæk 2010) where a group of end users act as domain experts in the conduction of Coorporative Usability Tests. That study shows that test output was enriched by including domain experts in the interpretation phase as they provided additional insight in identified problems and helped in uncovering a considerable amount of new problems (Følstad and Hornbæk 2010). The high performance of the software development practitioners within our study reveals that they have an elaborate understanding of the users of the system. This differs from the typical developer mindset presented in existing literature where it is reported that developers without usability knowledge have difficulties understanding their users, cf. (Ardito et al. 2011; Bak et al. 2008).

In terms of severity categorization we found a considerable disagreement between the ratings given by the practitioners and specialists. In the first test there was a disagreement in categorization in 70 % of the problems found by both groups where the practitioners consistently gave lower severity ratings than the specialists. During the second test there were disagreements in 22 % of the problems, giving an average of 46 % of all problems across both tests. This finding could indicate a downside to letting the practitioners test their own systems, as they may be subjectively biased. This is supported within existing literature, e.g. in (Rubin and Chisnell 2008) where it is established that development teams for objectivity reasons should not test their own designs. Although the practitioners' objectivity could be questioned in our case, we did find that they managed to uncover more problems than the specialists who had not taken part in the design or development of the system. A similar finding is presented in (Wright and Monk 1991) where it is shown that participants found more problems within their own designs than those made by others. Thus, the possible implications of objectivity may have a higher influence on severity ratings than on the number of problems identified.

Considering downstream utility we found that the practitioners before implementation primarily committed to fixing problems based on the factors of resource requirements and that severity ratings and frequency (no. of users experiencing a problem) did not matter. This prioritization resulted in a committed impact ratio of 61 %. Opposite findings are reported in (Law 2006), where it is found that severity ratings and frequency were the most influential factors on the impact ratio while the amount of resources required to fix the problems was less influential. This difference could be attributed to the fact that two of the practitioners in our study also had project management responsibilities, which could have caused an increased focus on resource requirements.

Considering downstream utility after implementing the second version of the system revealed no impact in terms of the number of identified usability problems. However,

the number of critical and serious problems was reduced considerably as the count was almost halved, which indicates a highly positive impact on system usability. On the other hand, the number of cosmetic problems was almost doubled. The increase in cosmetic problems could be explained by the critical and serious problems, which may have masked the cosmetic ones during the first test.

Although the number of identified problems remained on the same level, we found a completed-to-date impact ratio (CDIR) of 64 % as 21 of the 33 problems identified by the practitioners in the first test were removed. CDIR is also revealed within other studies utilizing user based tests, in these, however, usability practices were already established as specialists were involved in testing and redesigning the systems. The study conducted by Medlock and colleagues revealed a higher CDIR of 97 % by applying the RITE method (Medlock et al. 2002). In (Hertzum 2006) the average CDIR is 65 %, which was obtained through the conduction of 5 user based tests. In (Law 2006) usability specialists conducted what is mentioned as "standard user tests" based on video analysis. In that study the CDIR was 38.3 %. Thus, the CDIR of 64 % found within our study resembles the one presented in (Hertzum 2006), which was obtained from a company with established usability practices and with employed usability specialists. On the other hand, this finding is lower than that reported in (Medlock et al. 2002). This could be explained by the fact that each team member in the Medlock study had limited responsibilities as e.g. usability engineer or developer. The practitioners in our case had more responsibilities besides conducting the usability tests, e.g. writing new code, fixing functionality problems and project management responsibilities.

Taken together, the above findings regarding downstream utility reveal that the software development practitioners within our study prioritized fixing the majority of usability problems as well as eliminating most of these in the following implementation. This indicates that their mindsets deviate from the typical developer mindset as reported in (Ardito et al. 2011; Bak et al. 2008) where developers did not accept or prioritize fixing problems. This deviation from the typical developer mindset is also supported by their high performance in identifying problems as mentioned previously as this indicates a high level of user understanding.

CONCLUSION

The purpose of this study was to examine whether training software development practitioners in conducting usability tests could be a viable solution to overcome the barrier of developer mindset. We found that the practitioners after receiving 30 hours of training were able to identify an average of 80.5 % of all problems across the two tests conducted, which indicates a high level of user understanding. Their performance was slightly higher than that of the usability specialists who uncovered an average of 74.5 %. A reason for this may be the practitioners' level of domain knowledge, which proves to be another advantage of letting them conduct usability tests. We also found that the practitioners committed to fixing 61 % of the identified problems and that they managed to eliminate 64 %, which resembles impact ratios found in other settings where usability practices already have been established. These impact ratios indicate that the practitioners accepted and prioritized most of the problems, which deviates from the typical developer mindset found throughout existing literature.

The empirical data within our study is based on three practitioners from the same company conducting two tests, which makes it exploratory in nature. Further studies using more software development practitioners conducting tests are needed to validate our findings. Another limitation is that only one of the authors conducted an open coding analysis og the qualitative data where a more rigorous analysis would enrich and nuance findings of this study. Also, we have focused exclusively on training practitioners to conduct usability tests. As Wixon points out, then it is equally important to tell the practitioners what to do and not just what is wrong within an interface (Wixon 2003). Thus, in the future it would be crucial to provide such practitioners with training in interaction design to further increase the impact of usability tests.

ACKNOWLEDGMENTS

[Blind review]

REFERENCES

Ardito, C., Costabile, M.F., De Angeli, A., Lanzilotti, R.: Systematic evaluation of e-learning systems: an experimental validation. In Proc. NordiCHI, ACM Press, New York (2006).

Ardito C., Buono P., Caivano D., Costabile M.F., Lanzilotti R., Bruun A. and Stage J.: Usability Evaluation: A Survey of Software Development Organizations. In Proc. SEKE, Knowledge Systems Institute Graduate School, Skokie, IL, USA (2011).

Bak, J. O., Nguyen, K., Risgaard, P. and Stage, J. Obstacles to usability evaluation in practice: a survey of software development organizations. In Proc. NordiCHI, ACM Press (2008).

Bruun, A. and Stage, J.: Evaluating the Usability of Home Healthcare Applications. In Carsten Röcker and Martina Ziefle (Eds.), Human Centered Design of E-Health Technologies, IGI Global, Hershey, PA, USA (2010).

Bruun, A. Training Software Developers in Usability Engineering: A Literature Review. In Proc. NordiCHI, ACM Press, New York, NY, USA (2010).

Bruun, A. and Stage, J. 2011. Training Software Development Practitioners in Usability Evaluations: An Exploratory Study of Cross Pollination. In Proc. PUX (2011).

Frøkjær, E., Lárusdòttir, M.K.: Prediction of Usability: Comparing Method Combinations. In Proc. IRMA, Idea Group Publishing, Hershey (1999).

Følstad, A., Hornbæk, K. Work-domain knowledge in usability evaluation: Experiences with Cooperative

Usability Testing. In J. of Systems and Software 83, 11, 2019-2030, Elsevier Science Inc. New York, NY, USA (2010).

Gulliksen, J., Boivie, I., Persson, J., Hektor, A. and Herulf, L. Making a difference: a survey of the usability profession in Sweden. In Proc. NordiCHI, ACM Press (2004).

Häkli, A. Introducing user-centered design in a small-size software development organization, Helsinki University of Technology (2005).

Hertzum, M. Problem Prioritization in Usability Evaluation: From Severity Assessments Toward Impact on Design. In Int. J. of Human-Computer Interaction 21, 2, 125-146, Taylor & Francis (2006).

Howarth, J. Supporting novice usability practitioners with usability engineering tools, Virginia Polytechnic Institute & State University Blacksburg, Blacksburg, (2007).

Howarth, J., Andre, T.S., Hartson, R. A Structured Process for Transforming Usability Data into Usability Information. In Journal of Usability Studies 3, 1, 7-23, UPA (2007).

Høegh, R. T., Nielsen, C. M., Overgaard, M., Pedersen, M. B., Stage, J.: The Impact of Usability Reports and User Test Observations on Developers' Understanding of Usability Data: An Exploratory Study. In Int. Journal of Human-Computer Interaction 21, 2, 173-196, Taylor & Francis (2006).

Jacobsen, N.E., Hertzum, M., John, B.E.: The Evaluator Effect in Usability Studies: Problem Detection and Severity Judgments. In Proc. of HFES, HFES, Santa Monica (1998).

Juristo, N., Moreno, AM and Sanchez-Segura, M. I. Guidelines for eliciting usability functionalities. IEEE Transactions on Software Engineering 33, 11, 744-758, IEEE Computer Society Press (2007).

Kjeldskov, J., Skov, M. B. and Stage, J. Instant data analysis: conducting usability evaluations in a day. In Proc. NordiCHI, ACM Press (2004).

Koutsabasis, P., Spyrou, T., Darzentas, J.S., Darzentas J.: On the Performance of Novice Evaluators in Usability Evaluations. In Proc. PCI, Patra (2007).

Law, E. Evaluating the downstream utility of user tests and examining the developer effect: A case study. In Int. J. Human-Comput. Interact. 21, 2, 147-172, Taylor & Francis, London, England (2006).

Medlock, M.C., Wixon, D., Terrano, M., Romero, R., Fulton, B. Using the RITE method to improve products: a definition and a case study. In Proc. Usability Professionals Association, UPA (2002).

Metzker, E., Offergeld, M.: An Interdisciplinary Approach for Successfully Integrating Human-Centered Design Methods Into Development Processes Practiced by Industrial Software Development Organizations. In Proc. Int. Conf. on Engineering for Human-Computer Interaction, Springer-Verlag, London (2001).

Molich, R. User-Friendly Web Design, Ingeniøren Books, Copenhagen, Denmark (2000).

Nielsen, J. Iterative User-Interface Design. In Computer 26, 11, 32-41, IEEE (1993).

Nielsen, J. Finding usability problems through heuristic evaluation. In proc. CHI 1992, ACM Press, New York, USA (1992).

Nielsen, J. Usability inspection methods. In Proc. CHI 1994, ACM Press (1994).

Rubin, J., Chisnell, D.: Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests, 2nd. Edition, John Wiley & Sons, Inc., Indianapolis, USA (2008).

Sawyer, P., Flanders, A. and Wixon, D. Making A Difference – The Impact of Inspections. In Proc. CHI, ACM Press, New York, USA (1996).

Scholtz, J., Laskowski, S. and Downey, L. Developing usability tools and techniques for designing and testing web sites. In Proc. Conference on Human-Factors & the Web 1998.

Skov, M. B. and Stage, J., 2004. Integrating Usability Design and Evaluation: Training Novice Evaluators in Usability Testing. In Proc. of the Workshop on Improving the Interplay between Usability Evaluation and User Interface Design, ACM Press, New York (2004).

Skov, M. B., Stage, J.: Supporting Problem Identification in Usability Evaluations. In Proc. OZCHI '05. Computer-Human Interaction Special Interest Group (CHISIG) of Australia, Narrabundah (2005).

Skov, M. B. and Stage, J.: Direct Integration: Training Software Developers and Designers to Conduct Usability Evaluations. In Proc. of the First Workshop on the Interplay between Usability Evaluation and Software Development, CEUR-WS.org (2008).

Strauss, A., & Corbin, J. Basics of qualitative research. Techniques and procedures for developing grounded theory (2nd ed.), Thousand Oaks, CA: Sage (1998).

Wixon, D. Evaluating usability methods: why the current literature fails the practitioner. In Interactions 10, 4, 28-34, ACM New York, NY, USA (2003).

Wright, P.C., Monk, A.F. The Use of Think-Aloud Evaluation Methods in Design. In ACM SIGCHI Bulletin archive 23, 1, 55 – 57, ACM Press, New York (1991).

Zellweger, P.T., Bouvin, N.O., Jehøj, H., and Mackinlay, J.D. Fluid Annotations in an Open World. Proc. Hypertext 2001, ACM Press (2001), 9-18.

Let Your Users Do the Testing: A Comparison of Three Remote Asynchronous Usability Testing Methods

Anders Bruun[1], Peter Gull[2], Lene Hofmeister[3], Jan Stage[4]
[1]Mjølner Informatics A/S, Finlandsgade 10, DK-8200 Århus N, Denmark
[2]Jyske Bank A/S, Vestergade 8-16, DK-8600 Silkeborg, Denmark
[3]Nykredit A/S, Fredrik Bajers Vej 1, DK-9220 Aalborg East, Denmark
[4]Aalborg University, Department of Computer Science, DK-9220 Aalborg East, Denmark
ab@mjolner.dk, pg@jyskebank.dk, leho@nykredit.dk, jans@cs.aau.dk

ABSTRACT
Remote asynchronous usability testing is characterized by both a spatial and temporal separation of users and evaluators. This has the potential both to reduce practical problems with securing user attendance and to allow direct involvement of users in usability testing. In this paper, we report from an empirical study where we systematically compared three methods for remote asynchronous usability testing: user-reported critical incidents, forum-based online reporting and discussion, and diary-based longitudinal user reporting. In addition, conventional laboratory-based think-aloud testing was included as a benchmark for the remote methods. The results show that each remote asynchronous method supports identification of a considerable number of usability problems. Although this is only about half of the problems identified with the conventional method, it requires significantly less time. This makes remote asynchronous methods an appealing possibility for usability testing in many software projects.

Author Keywords
Remote testing, asynchronous testing, usability testing, empirical study.

ACM Classification Keywords
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Evaluation/methodology, Theory and methods.*

INTRODUCTION
User-based testing has become almost a de facto standard in usability engineering. The essential idea is to base assessment of the usability of a system on observation of users working with the system. The classical approach was to combine this idea of user-based testing with the think-aloud protocol and implement it in a laboratory setting, e.g. [27]. A key drawback of this approach was that it turned out to demand considerable resources for planning the tests, establishing a test setting and conducting the tests [8, 9, 22, 23, 24]. A subsequent analysis to identify usability problems, conducted as a rigorous walk-through of the hours of video recordings that documented the users' interaction with the system, was almost equally demanding in terms of resources [17].

The classical approach to usability testing has limited influence on contemporary practice in software development and commercial usability testing. Practitioners and researchers have developed a rich inventory of techniques for usability testing that reduce the resource demands, either for the whole evaluation process, e.g. [22, 23, 24], or for selected activities, e.g. [17]. Modern software organizations combine such techniques with new development approaches, e.g. [2]. At the other end of the spectrum, there are many software organizations that still have no systematic usability activities deployed in their development processes, and the most frequent cause given is perceived resource demands [3].

Software organizations that develop and evaluate products for global markets or practice outsourcing or global software development face different but equally significant obstacles. When developers, evaluators and users are distributed across different organizations, countries and time zones, user-based usability testing, in particular planning and setting up the test, becomes a nearly insurmountable logistic challenge.

The difficulties with usability testing that are emphasized above have led some researchers to inquire into remote usability testing. Remote usability testing denotes a situation where "the evaluators are separated in space and/or time from the users" [7]. There is a basic distinction between remote synchronous and remote asynchronous methods.

With a remote synchronous method, the test users and evaluators are separated in space [6]. The evaluators can observe the users over a network connection by means of video capture software where, for example, the content of the test participants' screen is transmitted to the evaluators who are residing in a remote location [1, 2, 6, 9, 12, 14]. A

CHI 2009, April 4–9, 2009, Boston, MA, USA.

remote synchronous method still requires the evaluators to be present in real time to control and monitor the test, and the results need to be analyzed by the evaluators. Thus, this method is almost as resource demanding as a typical user-based test, but it escapes many of the logistic problems [9].

With a remote asynchronous method, the test users and evaluators are separated both in space *and* time [6]. This implies that the evaluators no longer need to be present at the time when the users are working with the system. Thereby, the asynchronous methods eliminate a key drawback of the synchronous methods. Moreover, some of the asynchronous methods involve the users in the reporting of usability problems with the aim of also eliminating a detailed analysis conducted by the evaluators.

In this paper, we present results from an empirical study of three methods for remote asynchronous usability testing. In the following section, we describe previous research on remote asynchronous usability testing. Based on this, we have selected three asynchronous methods. Next, we describe the experimental method of the empirical study we conducted to compare the qualities of the three methods. This is followed by a presentation of the results of the study. Here, we focus on the number of usability problems identified and the time spent conducting the usability tests. Then we compare with results obtained by others and discuss our results. Finally, we provide the conclusion.

RELATED WORK

We have conducted a systematic study of literature on remote asynchronous usability testing. In this section, we provide an overview of papers that present empirical studies of the use of one or more remote asynchronous methods for usability testing. Thus papers that only mention or briefly outline remote asynchronous usability testing are not included. Table 1 provides an overview of the 22 papers.

Basis for comparison	Paper #
Conventional laboratory	0, 6, 19, 25, 26, 31, 32, 34, 35, 37, 38
Usability expert inspection	5, 6, 7, 13, 14, 29, 30, 38
No comparison	10, 16, 20, 28, 36

Table 1. Empirical papers with remote asynchronous usability testing methods and the comparisons they make.

There are 17 out of the 22 papers that compare remote asynchronous usability testing with established approaches such as a conventional user-based laboratory test, an expert-based usability inspection or both. Out of these, only one compares multiple asynchronous methods [6]. This is, however, only a cost-benefit comparison graph based on intuition and not empirical data. The other 16 papers only deal with a single remote asynchronous method. The remaining 5 papers are documenting empirical studies that apply an asynchronous method, but without comparing it to other approaches.

Table 2 provides an overview of the methods that are used in the papers listed in Table 1.

Method	Paper #
Auto logging	20, 28, 29, 30, 34, 35, 36, 37
Interview	10, 26, 29, 30, 35
Questionnaires	10, 19, 25, 28, 29, 30, 32, 34, 35, 36, 37
User-reported critical incident	0, 5, 6, 7, 13, 14, 31
Unstructured problem reporting	10, 19, 38
Forum	21
Diary	31

Table 2. Methods for remote asynchronous usability testing.

Auto logging is a method where quantitative data like visited URL history and the time used to complete tasks are collected in log files that are analysed. This method can show if the paths to complete tasks are well designed [20, 28, 29, 30, 34, 35, 36, 37], but it is lacking the ability to collect qualitative data needed to address usability issues beyond the likes of path finding and time used. Therefore, it is often combined with interviews and/or questionnaires as follow-up. With this combination, auto logging has found many of the same problems as heuristic inspection [30]. In another study, the evaluators used this approach to identify 60% of the problems found via a heuristic inspection [29]. In a comparison with a conventional laboratory method, it is concluded that the auto logging method in comparison is not too efficient [37]. The reports of two of these studies [30, 37] do not provide any information about the total number of usability problems identified with auto logging method. In another comparison, the evaluators using the auto logging method identified 40% of the usability problems found in a conventional laboratory test [34].

The *User-reported Critical Incident* method (UCI) is based on the idea that the users themselves report the problems they experience. This should relieve evaluators from conducting tests and analysing results. It has been concluded that test participants are able to report their own critical incidents, e.g. Castillo shows that a minimalist approach works well for training participants in identifying critical incidents [6]. The first studies of this method concluded that the participants were able to categorize the incidents [6, 7], but a more recent and systematic study concludes that the participants cannot categorize the severity of the critical incidents they identify [1]. The reason for this discrepancy may be that the training conducted by [5, 6, 7] was more elaborate than [1], but the training was done with the researchers physically present [6, 7, 31] which contradicts the idea of remote testing. The training conducted by [1] was done over the Internet. The number of usability problems identified also varies between the different studies. In one of the first studies, 24 participants identified 76% of the usability problems found by experts [6]. In a later study, 10 participants identified

60% of the problems found in a conventional laboratory test [31]. In the most recent study, 6 participants identified 37% of the usability problems found in a conventional laboratory test [1].

Unstructured problem reporting is based on the idea that participants make notes on the usability problems they encounter while working on a set of tasks [10, 19, 38]. The descriptions of the studies of this method are brief and there is no information about the predetermined content they have wanted participants to write down. In a study, 9 participants using this kind of reporting identified 66% of the usability problems found by experts [38]. These researchers recommend a more structured approach which is close to the user-reported critical incident method. Another study showed that 8 participants using the unstructured approach identified 50% of the total usability "issues" found in a conventional laboratory test, but this was based on a procedure where the participants in the remote asynchronous condition solved instructional tasks whereas the participants in the laboratory test solved exploratory tasks. As noted by the authors, this makes the comparison "unfair" [19].

The *forum* has been used as a source for collecting qualitative data in a study of auto logging [20]. The researchers did not specifically encourage the participants to report usability issues through the forum, but the participants did report detailed usability feedback. There is no information about user training or the number of usability problems reported in the forum [20]. In a different study, the author argues that participants may be more motivated to report problems, if reporting is a collaborative effort amongst participants. The author believes that participants through collaboration may give input which increases data quality and richness compared to the user-reported critical incident method [31].

The *diary* has been used on a longitudinal basis for participants in a study of auto logging to provide qualitative information [30]. There is no information about the usefulness of the method or the experiences with it. However, it is mentioned that the participants used their diary on a longitudinal basis to report on the usability problems they experienced with the use of a particular hardware product.

METHOD

We have conducted an empirical study of remote asynchronous usability testing with the following four conditions:

- Conventional user-based laboratory test (Lab)
- User-reported critical incident (UCI)
- Forum-based online reporting and discussion (Forum)
- Diary-based longitudinal user reporting (Diary)

The conventional laboratory test was included to serve as a benchmark. In the rest of this section, we describe the method of the study.

Participants. A total of 40 test subjects participated, ten for each condition. Half of the participants were female and the other half male. All of them studied at Aalborg University, at different faculties and departments. They were between 20 and 30 years of age. They signed up voluntarily after we submitted an email call for participants. Half of them were taking a non-technical education (NT), and the other half a technical education (T). For all four test conditions the participants were distributed as follows: 3 NT females, 2 T females, 2 NT males and 3 T males. Most of the participants reported medium experience in using IT in general and an email client. Two participants reported themselves as being beginners to IT in general and had medium knowledge of using an email client. None of the participants had previous knowledge about usability testing. They received no course credit or payment. After completion, we gave each a bottle of wine for their effort.

Training. The test subjects participating in the remote asynchronous sessions were trained in identification and categorisation of usability problems. This was done using a minimalist approach that was strictly remote and asynchronous, as they received written instructions via email, explaining through descriptions and examples what a usability problem is and how it is identified and categorised. The categories were "low", "medium" and "high", corresponding to the traditional cosmetic, serious and critical severity ratings [1].

System. In order to facilitate comparison of the results, we chose to use the same system and tasks as another study that involved both synchronous and asynchronous methods [1]. Accordingly, we tested the email client Mozilla Thunderbird version 1.5. None of the test subjects had used Mozilla Thunderbird before.

Tasks. All participants were asked to solve the following tasks (the same as the ones used by [1]):

1. Create a new email account (data provided)
2. Check the number of new emails in the inbox of this account
3. Create a folder with a name (provided) and make a mail filter that automatically moves emails that has the folder name in the subject line into this folder
4. Run the mail filter just made on the emails that were in the inbox and determine the number of emails in the folder
5. Create a contact (data provided)
6. Create a contact based on an email received from a person (name provided)
7. Activate the spam filter (settings provided)
8. Find suspicious emails in the inbox, mark them as spam and check if they were automatically deleted
9. Find an email in the inbox (specified by subject line contents), mark it with a label (provided) and note what happened

We had fixed tasks across the four conditions to ensure that all participants used the same parts of the system.

Laboratory Testing (Lab)

Setting. The laboratory test was conducted in a state-of-the-art usability laboratory. In the test room, the test participant sat in front of the computer and next to her/him sat a test monitor whose primary task was to make sure that the test participant was thinking aloud.

Procedure. The procedure followed the guidelines of [27]. It was not conducted by the authors of this paper. The test subjects were introduced to the test sequence and the concept of thinking aloud by the test monitor. We scheduled one hour per participant including post-test interview and switching participants. The interviews were carried out by the test monitor. Participants had to solve the nine tasks while thinking aloud. The test monitor had a timeframe for the completion of each task. Participants who had not solved a task in this time received help from the test monitor to ensure that all tasks were completed.

Data collection. A video of the test subjects' desktop was recorded along with video showing the test subject's face.

This condition was referred to in the introduction as the classical approach. We decided to use it as a benchmark, despite its limited influence on contemporary usability testing practice, because it facilitates comparison with other studies, where it is commonly used. The condition involved thinking aloud, which in discussed in [10]. Again, we used it to facilitate comparison with other studies.

The Three Remote Conditions

Some methodological aspects were common to all three remote conditions.

Setting. In all remote asynchronous methods, participants worked at home using their own computer. The participants could carry out the tasks whenever they wanted; but had to completed by a specified date. Once they started, they had to finish all tasks in one session.

Procedure. We sent all participants the training material and a guide on how to install the system. The participants were asked first to go through the training material, install Mozilla Thunderbird and then begin task completion. With each task there was a hint that allowed the participant to check whether they had solved the task correctly. In the Lab condition, you can control the users' task solving process in accordance with the correctness of their solution. Remote users need a similar criterion in order to know when they can stop the work on a task. That is the purpose of the hint.

User-Reported Critical Incident Method (UCI)

Procedure. The participants were instructed to report any negative critical incident they might find both major and minor, as soon as they discovered it. This was done using a web based report form that was programmed using PHP, JavaScript and an MySql database. The content of the form was similar to that used by [5] and [31]. The following questions had to be answered using this form:

- What task were you doing when the critical incident occurred?
- What is the name of the window in which the critical incident occurred?
- Explain what you were trying to do when the critical incident occurred.
- Describe what you expected the system to do just before the critical incident occurred.
- In as much detail as possible, describe the critical incident that occurred and why you think it happened.
- Describe what you did to get out of the critical incident.
- Were you able to recover from the critical incident?
- Are you able to reproduce the critical incident and make it happen again?
- Indicate in your opinion the severity of this critical incident.

The participants were also asked to create a log of time spent completing each task and email this log to us.

Data collection. At the bottom of the online form was a submit button. When it was pressed, the data was saved in an online database and the form was reset, ready to enter a new entry. The form was running in a separate browser window, requiring the participants to toggle between windows when they encountered a problem. Reporting might be integrated directly into the application [13], but it requires extra resources to implement, and the two-window approach has been shown to work as well [31].

Forum

Procedure. After installing the system, the participants were asked to first take notes on the usability problems they experienced during completion of the tasks and also to rate the severity. The participants were asked to finish all tasks in one sequence and to create a log of the time taken to finish each task. After completion of the tasks the test participants were instructed to uninstall Mozilla Thunderbird to avoid confounding longitudinal use. They were then asked to post and discuss their experienced usability problems with the other participants. They were given a week for that. Each participant was given the following instructions: A) Check if the given usability problem already exists. B) If the problem does not exist then add a problem description and a severity categorization. C) If the problem is already mentioned, comment on this either by posting an agreement with the problem description and categorization or state a disagreement with a reason.

Data collection. The forum in itself is a data collection tool, so the data collection for this method was very simple.

Diary

Procedure. The participants were given a timeframe of five days to write about experienced usability problems and severity categorizations in their diary. We provided the same list of elements to consider as to those participating in the forum condition. They were also asked to create a time

log over the time taken to finish each task. We did not impose any formal content structure. On the first day, the participants received the same nine tasks as all other participants. We instructed them to complete those nine tasks on the first day, and then email the experienced problems and the log to us immediately after completion.

During the remaining four days the participants received additional tasks to complete on a daily basis. These tasks were of the same types as the original nine tasks. The purpose was to generate longitudinal data by ensuring that the users kept working with the system through all five days and making daily entries in their diary. The longitudinal element was only introduced after the first day to ensure that the data from that day were comparable with the other conditions and the value of more days could be identified.

Data collection. The participants e-mailed their diary notes to us after the first and the fifth day.

Data Analysis

The data analysis was conducted by three of the authors of this paper. Each analyzed all data from all four test conditions. This consisted of 40 data sets, 10 for each condition. All data was collected before conducting the analysis. Each data set was given a unique identifier, and a random list was generated for each evaluator, defining the order of analysis of all data sets. Each evaluator analyzed all the data sets alone, one at a time.

For the Lab condition, the videos were thoroughly walked through. The data from the three remote conditions was read one problem at a time. By using only the information available in the users' problem description, it was transformed into a usability problem description. If necessary, Mozilla Thunderbird was checked to get a better understanding of the problem. When analyzing forum descriptions, previous problem descriptions by other users in the same forum thread was also included in the analysis. If a description could not be translated into a meaningful problem in short time or we could not identify the problem using Thunderbird, the problem was not included in the problem list, because we wanted to make sure it reflected the users' experience and not problems made up by the evaluators. There were 12 user descriptions (1 from UCI, 3 from Forum and 8 from Diary) that could not be translated into a problem description because the description was impossible to understand or missing.

During the analysis we also rated the severity of the problems, as we wanted to make sure that this was done consistently.

When each evaluator had created a problem list for all data sets, they merged their lists for each of the four conditions. These four lists were then merged to form a complete problem list for the individual evaluator. The three evaluators then merged their individual lists for each of the four conditions in order to create a joint problem list for that condition. In case of disagreement, they negotiated by referring to the system and the original data until they reached an agreement. Severity rating in the joined lists was done using the most serious categorization. The joined lists for each condition were then joined to form a complete joined problem list. The resulting problem list included a detailed description of each usability problem.

	Lab	UCI	Forum	Diary	Avg.
Problems agreed on	23.3	9	8	17.7	14.5
Number of problems	46	13	15	29	25.8
Any-two agreement	50.7%	69.2%	53.3%	60.9%	56.3%

Table 3. The average any-two agreement between the evaluators for all test conditions.

Hertzum and Jacobsen [15] have shown that evaluators do not find exactly the same usability problems from the same data set. They call this the evaluator effect. To verify the agreement between evaluators, the evaluator effect can be calculated as an any-two agreement showing to what extent the evaluators have identified the same problems [15]. Table 3 shows the average any-two agreement for all of the test conditions and for the entire test. Compared to Hertzum and Jacobsen's findings [15], our any-two agreement is very high.

RESULTS

This section presents the findings from the study.

	Lab N=10		UCI N=10		Forum N=10		Diary N=10	
Task completion time in minutes: Average (SD)	24.24 (6.3)		34.45 (14.33)		15.45 (5.83)		*Tasks 1-9:* 32.57 (28.34)	
Usability problems:	**#**	**%**	**#**	**%**	**#**	**%**	**#**	**%**
Critical (21)	20	95	10	48	9	43	11	52
Serious (17)	14	82	2	12	1	6	6	35
Cosmetic (24)	12	50	1	4	5	21	12	50
Total (62)	46	74	13	21	15	24	29	47

Table 4. Number of identified problems and task completion time using the Lab, UCI, Forum and Diary methods. Percentages are of the total number of problems shown in brackets in the left column.

Number of Problems Identified and Time Spent

In this section we compare the four conditions with respect to the number of usability problems identified and the time spent on analysis. There is considerable variation in the standard deviations between the conditions. As we have no data on the task solving process in the remote conditions, we cannot explain this variation. An overview of the problems identified can be seen in table 4.

Table 5 shows the time spent in the four conditions. All time indications are the sum for all evaluators involved in that activity. The time for preparation does not include time

spent on finding test subjects, as this was done jointly for all four conditions. In total, it took about 8 hours. Task specifications were taken from an earlier study. Preparation in the three remote conditions was primarily to work out written instructions for the participants. These instructions could to a large extent be reused between the conditions, thus a test with only a single remote method would require a few hours more.

	Lab (46)	UCI (13)	Forum (15)	Diary (29)
Preparation	6:00	2:40	2:40	2:40
Conducting test	10:00	1:00	1:00	1:30
Analysis	33:18	2:52	3:56	9:38
Merging problem lists	11:45	1:41	1:42	4:58
Total time spent	61:03	8:13	9:18	18:46
Avg. time per problem	1:20	0:38	0:37	0:39

Table 5. Person hours spent on test activities. The numbers in parentheses are the total number of problems identified.

The UCI condition included setting up a web based form that the participants should use to report each problem they identified. This was made by first developing a tool and then using this tool to create the form. This took about 16 hours, but the tool and most of the form is directly reusable for a new test, which is why we have excluded this time.

The time for conducting the test in the Lab condition is the time spent by the test monitor. This includes the time it took the user to solve the tasks as well as setting up the system, briefing the user and administering a questionnaire.

	Lab	UCI	Forum	Diary
Lab		p<0.001 ***	p<0.001 ***	p=0.0031 **
UCI	p<0.001 ***		p=0.6639	p=0.002 **
Forum	p<0.001 ***	p=0.6639		p=0.01423 *
Diary	p=0.0031 **	p=0.002 **	p=0.01423 *	

Table 6. Fisher's exact test for the total number of usability problems identified in the four conditions. * = significant difference, ** = Very significant difference, * = Extremely significant difference**

Lab

From the Lab test we identified a total of 46 usability problems. Twenty of these were critical, 14 serious and 12 cosmetic. Comparing this result to the total of 62 problems, we were able to identify 74% of all reported problems using the Lab condition. As many as 95% of the critical problems, 82% of the serious and 50% of all cosmetic problems were found using this method. Thus the Lab condition identified more problems than the others, but at the same time it was the most time consuming as we spent 61 hours on it.

Lab vs. UCI

The UCI condition revealed 13 problems, consisting of 10 critical, 2 serious and 1 cosmetic. A Fisher's exact test gives an extremely significant difference (see table 6 for an overview) meaning that the Lab condition identified more problems than UCI. Fisher's exact test for each level of severity gives p=0.00139 for critical problems, p<0.001 for serious and p<0.001 for cosmetic. Thus for each severity level, the Lab condition finds more problems than UCI.

There was some overlap between the problems identified in the two conditions. All the 10 critical problems found with UCI were also found with the Lab condition. One of the 2 serious problems found with UCI was also found in the Lab condition. The single cosmetic problem found with UCI was not found in the Lab condition.

The UCI condition was clearly less time-consuming as we only spent just over 8 hours compared to the 61 hours for the Lab condition.

Lab vs. Forum

With the Forum condition we could identify a total of 15 problems (9 critical, 1 serious and 5 cosmetic). A Fisher's exact test reveals an extremely significant difference (p<0.001), meaning that the Lab condition identified significantly more problems than the Forum condition. For the individual severity levels, Fisher's exact test gives p<0.001 for the critical, p<0.001 for the serious and p=0.0687 for the cosmetic problems. Thus the Lab condition is significantly better for the critical and serious problems, but no significant difference for cosmetic ones.

In the Lab condition we found all the 9 critical problems that were identified by the Forum. The serious problem from the Forum condition was also identified in the Lab condition, and 3 of the 5 cosmetic problems were also in common between the Lab and Forum.

We spent just over 9 hours on the Forum condition compared to the 61 hours on the Lab condition.

Lab vs. Diary

The Diary condition revealed a total of 29 problems, consisting of 11 critical, 6 serious and 12 cosmetic. A Fisher's exact test shows that the Lab condition identified significantly more problems than the Diary condition. For the severity levels, the Lab condition found significantly more critical (p=0.0036) and serious problems (p=0.013), whereas there was no significant difference for the cosmetic problems (p=1.00).

Out of the 11 critical problems found with the diary condition, 9 were also revealed by the Lab condition. For the serious problems, 3 of the 6 Diary problems were also found in the Lab condition. Finally, 3 of the 12 cosmetic problems were also found with the Lab method.

The time spent on the Diary condition was just under 19 hours compared to the 61 hours in the Lab condition.

UCI vs. Forum

The UCI and Forum conditions have 5 critical problems in common and did not find the same serious or cosmetic problems. A Fisher's exact test reveals no significant difference (p=0.6639) in the total number of problems identified between the two methods. Thus we cannot identify any one of them as performing best overall. There is no significant difference for the severity levels either.

We spent almost the same time on the UCI condition, just over 8 hours, and Forum conditions, just over 9 hours.

UCI vs. Diary

Using the UCI and Diary conditions we found 7 critical problems and 2 serious problems common for both methods. On the total number of problems identified we found a very significant difference (p=0.002), meaning that the Diary condition performed better than UCI. For the individual severity levels, the only significant difference was on cosmetic problems, where the Diary condition found more problems (p<0.001).

We spent just under 19 hours on the Diary condition and just over 8 hours on the UCI condition.

Forum vs. Diary

With the Forum and Diary conditions we found 7 critical problems, 1 serious and 1 cosmetic problem common for both methods. The difference on the total number of problems identified is significant (p=0.0142), meaning that the Diary condition performed better than the Forum. For the three severity levels, there is no significant difference.

We spent just under 19 hours on the Diary condition compared to just over 9 hours on the Forum condition.

Task Completion

For all 40 test participants the mean value of completed tasks is 8.9 out of 9. The only condition, in which not all tasks were completed, was UCI, where one test participant did not complete tasks 3 and 4, because that person had no understanding of a filter and how it worked. All participants completed the 9 tasks in the Lab condition, but the majority of participants experienced difficulties in completing tasks 3, 6 and 7. For task 3 and 7, it was caused by limited or no understanding of filters, and for task 6, it was a difference compared to Outlook that confused some users.

Task Completion Time

Table 4 gives an overview of the average time spent completing all tasks. For tasks 1-9 the most significant difference is between the Forum and UCI conditions. Participants in the Forum spent 15.45 (SD=5.83) minutes completing all 9 tasks, whereas UCI participants spent 34.45 (SD=14.33) minutes. In between we find the Lab condition, in which participants spent 24.24 (SD=6.3) minutes and the Diary with 32.57 (SD=28.32) minutes.

The standard deviation emphasizes a considerable difference in the participants' completion time for the Diary condition compared to the other conditions. The completion times varied from a minimum of about 4 minutes to complete tasks 1-9 up to a maximum of 99 minutes.

Unique Problems

Some of the problems we have identified were only found in one condition. Table 7 gives an overview of the number of problems identified in one test condition only.

	Lab	UCI	Forum	Diary	Total
Critical (21)	5	0	0	1 / **1**	6
Serious (17)	11	1	0	2 / **0**	14
Cosmetic (24)	7	0	2	9 / **3**	18
Total (62)	23	1	2	12 / **4**	38

Table 7. The number of problems identified during one test condition only. The numbers in parentheses are the total number of problems for each categorization and the numbers in bold are the number of unique problems identified using the diary during the extra days of task solving.

From table 7 it is clear that the Lab test revealed many problems not found by any of the remote asynchronous conditions. 37% percent of the problems were identified only in the Lab condition. The majority of these are serious and 24% of all critical problems identified are only identified in the laboratory condition. Looking at all three severity categories, the unique Lab problems are primarily within the theme "Information", cf. [21], i.e. problems where the participants were missing information or did not understand the information from the system as it was too technical. The UCI and Forum conditions, also being the ones revealing the smallest number of problems in total, have revealed 3 unique problems in total, none of them being critical. The diary has revealed even more unique cosmetic problems than the laboratory condition (9). The unique problems found via the Diary condition are distributed evenly over the different problem themes defined in [21].

The Lab condition identified 5 critical problems not found by any of the remote methods. Interestingly, this means that by combining the results from the UCI, Forum and Diary conditions, we have identified 16 of the total of 21 critical problems. The total time for all three remote conditions sums up to about 36 hours, which is just over half of the time spent on the Lab condition.

Differences between Severity Ratings

All the users in the three remote conditions received the same instructions on rating of the severity of usability problems. We explained it to the users as categories, but they translate directly into a standard three-level severity rating scale. In this section, we examine whether it was possible for the participants from the remote conditions to categorise the problems properly. Table 8 shows how the participants' severity ratings corresponded to the ones made by the evaluators.

	UCI (13)		Forum (15)		Diary (29)	
	#	%	#	%	#	%
Same categorisation	10	77	10	66	13	45
No categorisation	0	0	4	27	11	38
Lower participant categorisation	1	8	1	7	2	7
Higher participant categorisation	2	15	0	0	3	10
Total	13	100	15	100	29	100

Table 8. Number of problems, where the participants' categorisations did or did not match those done by the evaluators.

The most structured method of the three, UCI, was the one where the severity ratings best matched the ones made by the evaluators. In this case, the ratings of only 3 of the 13 problems (23%) did not match, and all problems were rated. The reason for this was that it was not possible to report a problem without a severity rating. In the Diary condition, as many as 11 out of 29 (38%) problems were not rated. The Forum method is, like the Diary, a more unstructured approach than UCI. In this condition, 5 problems out of 15 (34%) were not categorized similarly, including problems without a rating.

Differences in Problems

We have seen a difference in the problem themes identified using the different methods. The critical problems identified using the asynchronous methods are primarily of the theme "User's Mental Model", cf. [21]. In the Lab condition many such problems are also identified. What is typical about these problems are, that the participants' logic is not consistent with the logic of the system. We can also see that the Lab condition has facilitated the identification of many "information" problems (13), as opposed to the asynchronous conditions. These problems are mainly concerning lack of information or information that is not understandable by the user and confirm the value of the think-aloud protocol in the Lab condition.

DISCUSSION

With the three remote asynchronous methods we have studied, detailed analysis of usability problems is replaced with an activity where descriptions made by the users are transformed into a list of usability problems. With UCI, this transformation was very simple. You could almost "copy-paste" several of the users' descriptions directly into the problem list. This also corresponds to the findings of others [6]. The reason is undoubtedly that the participants were forced to fill out certain fields and thereby provide specific information. The output from the forum also required very little work, but the quality differed considerably. The idea of the forum was to allow participants to discuss the problems collaboratively. Thereby, we hoped to achieve a richer description of each problem. This required actual discussion to take place, which was very limited. The problems that were discussed did, however, give a clearer understanding of especially what led to a given problem. The longitudinal aspect of the Diary condition was intended to give the participants a better basis for problem identification and reporting and enable them to identify problems that were only identifiable during longer use of the program. The problem descriptions did not improve over time and the extra four days only provided a total of 7 problems, only 4 of these being unique for the Diary condition. The unstructured nature of the diaries required a greater amount of interpretation resulting in a more pronounced evaluator effect and considerably more time spent on analysis.

The research literature on remote asynchronous usability testing is limited. Therefore, we have only few possibilities for comparison our results with others. The benefits of the UCI method have been identified before [6]. With this method, we identified only one cosmetic problem. A similar tendency is reported in [1]. On the number of problem identified with UCI, there are some remarkable differences. We found 28% of the problems found in the laboratory condition. This is close to a previous result of 37% [1], but very different from earlier results of 76% [6] and 60% [31]. This may be due to the difference in training, as we, like [1], have given the participants written instructions. [6] used video training and exercises as well, and [31] used an online training tool.

The only reference on the Forum method has no information about the number of problems identified and the resources spent [20]. In a recent study [18], 2 forums were used to evaluate the user experience with two different versions of the same game. The two forums worked out differently, with one producing more relevant and detailed information than the other. The reason seemed to be that they started out differently. Based on that study and the one reported in this paper, we would suggest the Forum method should be extended with a moderator to ensure a good start, enough details in the descriptions and ratings of severity.

Research on the Diary method is very limited. The single reference [30] we have found has no results or comments about how well the method performed. In our study, the Diary condition required more time for analysis. This could, however, be reduced if the diary was combined with the problem reporting format that is used with UCI, but that would change the whole idea of the diary.

Another difference between the Lab and the three remote asynchronous conditions is the training of users. We received very few useless descriptions from our users, but as emphasized above the number of problems identified was lower than those reported from some of the other studies. It would be interesting to experiment with the effect of different training formats and materials.

One of the difficulties in our study was that we did not observe the participants in the remote conditions. This is in

line with the philosophy of the methods, but the consequence is that we have missed information about their task solving process. It also means that the task completion times have to be read with great caution.

The number of usability problems identified is a key element in our comparison of the different conditions. This is typical in the research literature on method comparisons. It would be relevant to compare based on other measures, e.g. the method's ability to reveal the users' understanding of the system or the usefulness of the problems identified.

The Lab condition has been used as a benchmark, although it has limited influence in practice. The reason is to facilitate comparison with related work. It would be interesting to conduct a follow-up study where the remote conditions are compared to more modern approaches.

Our study was based on a system that was a finished product. A main challenge for usability engineering is to conduct testing early in the development process. This may be more difficult with a remote asynchronous method, because the users are on their own with no possibility of getting support with a system that is not fully functional.

CONCLUSIONS

This paper has reported from an empirical study of three remote asynchronous usability testing methods. The methods were compared to each other and to a classical laboratory-based approach. On the overall level, the three remote methods performed significantly below the classical lab test in terms of the number of usability problems identified. For critical and serious problems, the Diary condition, which was the best of the remote methods, identified only half of the problems found in the Lab condition. The other two remote methods performed similarly for critical problems but worse for serious problems. This may seem disappointing. Yet two of the remote methods produced these results with an effort that only amounted to about 13% of what the lab test took. The diary method took more time but still only about 30%.

This makes the remote methods an appealing possibility for many software projects. It is often highly relevant to get a cheap usability test although it is not complete. In that case, one of the remote tests would be an interesting possibility. In addition, the remote methods seemed to complement each other, thus a combination of two or all three is a cost-effective solution.

Our study is limited in a number of ways. The number of test subjects in each condition was only 10 persons. This number could have been higher, but it is quite typical for method experiments. The users in the study were university students. This may introduce a bias as they may be more used to make and report assessments.

The results convey a number of interesting directions for future work. First of all, it would be interesting to try the methods out in real software projects with ordinary users. A mere replication would also be highly relevant because of the limited amount of experimental data about remote methods. This could involve improvement of each method based on our experience. The basis for comparison could also be extended with a more modern approach where the test monitor is directly involved in the identification of usability problems.

ACKNOWLEDGMENTS

The research behind this paper was partly financed by the Danish Research Councils (grant number 2106-04-0022). We are very grateful to the 40 test subjects that helped us in the usability tests. We would also like to thank the anonymous reviewers for their comments and advice.

REFERENCES

1. Andreasen, M. S., Nielsen, H. V., Schrøder, S. O. and Stage, J. What happened to remote usability testing? An empirical study of three methods. *Proceedings of CHI 2007,* ACM Press (2007), 1405-1414.
2. Au, I., Boardman, R., Jeffries, R., Larvie, P., Pavese, A., Riegelsberger, J., Rodden, K., and Stevens, M. User experience at google: focus on the user and all else will follow. *Proceedings of CHI 2008,* ACM Press (2008), 3681-3686.
3. Bak, J. O., Nguyen, K., Risgaard, P., and Stage, J. Obstacles to usability evaluation in practice: a survey of software development organizations. *Proceedings of NordiCHI 2008*. ACM Press (2008), 23-32.
4. Brush, A. B., Ames, M., and Davis, J. A comparison of synchronous remote and local usability studies for an expert interface. *Proceedings of CHI 2004,* ACM Press (2004), 1179-1182.
5. Capra, M. G. *An Exploration of End-User Critical Incident Classification*. Master thesis, Virginia Polytechnic Institute and State University, 2001.
6. Castillo, J. C. *The User-Reported Critical Incident Method for Remote Usability Evaluation*. Master thesis, Virginia Polytechnic Institute and State University, 1997.
7. Castillo, J. C., Hartson, H. R. and Hix, D. Remote usability evaluation: Can users report their own critical incidents? *Proceedings of CHI 1998*, ACM Press (1998), 253-254.
8. Desurvire, Heather W. Faster, cheaper!! Are Usability Inspection Methods as Effective as Empirical Testing? John Wiley & Sons, 173-202, 1994.
9. Dray, S. and Siegel, D. Remote possibilities?: International usability testing at a distance. *interactions 11*, 2 (2004), 10-17.
10. Ericsson, K. A. and Simon, H. A. How to study thinking in everyday life: contrasting think-aloud protocols with descriptions and explanations of thinking. *Mind, Culture, and Activity 5*, 3 (1998), 178-186.

11. Følstad, A., Brandtzæg, P. B. and Heim, J. Usability Analysis and Evaluation of Mobile ICT Systems. http://www.hft.org/HFT01/paper01/mobility/25_01.pdf

12. Hammontree, M., Weiler, P. and Nayak, N. Remote usability testing. *Interactions 1*, 3 (1994), 21-25.

13. Hartson, H. R. and Castillo, J. C. Remote evaluation for post-deployment usability improvement. *Proceedings of AVI 1998*, ACM Press (1998), 22-29.

14. Hartson, H. R., Castillo, J. C., Kelso, J. and Neale, W. C. Remote evaluation: The network as an extension of the usability laboratory. *Proceedings of CHI 1996*, ACM Press (1996), 228-235.

15. Hertzum, M. and Jacobsen, N. E. The evaluator effect: A chilling fact about usability evaluation methods. *International Journal of Human-Computer Interaction 15*, 1 (2003), 183-204.

16. Hilbert, D. M. and Redmiles, D. F. Separating the wheat from the chaff in internet-mediated user feedback expectation-driven event monitoring. *ACM SIGGROUP Bulletin 20*, 1 (1999), 35-40.

17. Kjeldskov, J., Skov, M. B. and Stage, J. Instant Data Analysis: Evaluating Usability in a Day. *Proceedings of NordiCHI 2004*, ACM Press (2004), 233-240.

18. Larsen, J. M. *Playful Interaction*. Master thesis, Aalborg University, Department of Computer Science, 2008.

19. Marsh, S. L., Dykes, J. and Attilakou, F. Evaluating a geovisualization prototype with two approaches: remote instructional vs. face-to-face exploratory. *Proceedings of Information Visualization 2006*, IEEE (2006), 310-315.

20. Millen, D. R. Remote usability evaluation: user participation in the design of a web-based email service. *ACM SIGGROUP Bulletin 20*, 1 (1999), 40-45.

21. Nielsen, C. M., Overgaard, M., Pedersen, M. B., Stage, J. and Stenild, S. It's worth the hassle! the added value of evaluating the usability of mobile systems in the field. *Proceedings of NordiCHI 2006*, ACM Press (2006), 272-280.

22. Nielsen, J. Finding usability problems through heuristic evaluation. *Proceedings of CHI 1992*, ACM Press (1992), 373-380.

23. Nielsen, J. and Molich, R. Heuristic evaluation of user interfaces. *Proceedings of CHI 1990*, ACM Press (1990), 249-256.

24. Nielsen, J. Usability inspection methods. *Proceedings of CHI 1994*, ACM Press (1994), 377-378.

25. Olmsted, E. and Gill, M. In-person usability study compared with self-administered web (remote-different time-place) study: does mode of study produce similar results? *Proceedings of UPA 2005*, UPA (2005).

26. Petrie, H., Hamilton, F., King, N. and Pavan, P. Remote usability evaluation with disabled people. *Proceedings of CHI 2006*, ACM Press (2006), 1133-1141.

27. Rubin, J. *Handbook of Usability Testing*. John Wiley and Sons, 1994.

28. Scholtz, J. A case study: developing a remote, rapid and automated usability testing methodology for on-line books. *Proceedings of HICSS 1999*, IEEE (1999).

29. Scholtz, J. and Downey, L. Methods for identifying usability problems with web sites. *Proceedings of IFIP Conference,* ACM Press (1998), 191-206.

30. Steves, M. P. et. al. A comparison of usage evaluation and inspection methods for assessing groupware usability. *Proceedings of CSCW 2001*, ACM Press (2001), 125-134.

31. Thompson, J. A. *Investigating the Effectiveness of Applying the Critical Incident Technique to Remote Usability Evaluation*. Master thesis, Virginia Polytechnic Institute and State University, 1999.

32. Tullis, T., Fleischman, S., McNulty,M., Cianchette, C. and Bergel, M. An empirical comparison of lab and remote usability testing of web sites. http://home.comcast.net/~tomtullis/publications/RemoteVsLab.pdf

33. Vermeeren, A. P. O. S., van Kesteren, I. and Bekker, M. M. Managing the 'evaluator effect' in user testing. *Proceedings of INTERACT 2003*, IOS Press (2003).

34. Waterson, S., Landay, J. A. and Matthews, T. In the lab and out in the wild: remote web usability testing for mobile devices. *Proceedings of CHI 2002*, ACM Press (2002), 796-797.

35. West, R. and Lehman, K. R. Automated Summative Usability Studies: An Empirical Evaluation. *Proceedings of CHI 2006*, ACM Press (2006), 631-639.

36. Winckler, M. A. A., Freitas, C. M. D. S. and de Lima, J. V.Remote usability testing: a case study. *Proceedings of OzCHI 1999*, CHISIG (1999).

37. Winckler, M. A. A., Freitas, C. M. D. S. and de Lima, J. V. Usability remote evaluation for www. *Proceedings of CHI 2000*, ACM Press (2000), 131-132.

38. Äijö, R. and Mantere, J. Are Non-Expert Usability Evaluations Valuable? http://www.hft.org/HFT01/paper01/acceptance/2_01.pdf

The Effect of Task Assignments and Instruction Types on Remote Asynchronous Usability Testing

Anders Bruun
Dept. of Computer Science, Aalborg University
Selma Lagerlöfs Vej 300, DK-9220, Aalborg Oest
bruun@cs.aau.dk

Jan Stage
Dept. of Computer Science, Aalborg University
Selma Lagerlöfs Vej 300, DK-9220, Aalborg Oest
jans@cs.aau.dk

ABSTRACT
Remote asynchronous usability testing involves users directly in reporting usability problems. Most studies of this approach employ predefined tasks to ensure that users experience specific aspects of the system, whereas other studies use no task assignments. Yet the effect of using predefined tasks is still to be uncovered. There is also limited research on instructions for users in identifying usability problems. This paper reports from a comparative study of the effect of task assignments and instruction types on the problems identified in remote asynchronous usability testing of a website for information retrieval, involving 53 prospective users. The results show that users solving predefined tasks identified significantly more usability problems with a significantly higher level of agreement than those working on their own authentic tasks. Moreover, users that were instructed by means of examples of usability problems identified significantly more usability problems than those who received a conceptual definition of usability problems.

Author Keywords
Remote testing, asynchronous testing, usability testing, task assignments, instruction types, empirical study.

ACM Classification Keywords
H.5.2 [**Information Interfaces and Presentation**]: User Interfaces – *Evaluation/methodology, Theory and methods.*

INTRODUCTION

Remote usability testing has been promoted as a solution to key logistic challenges in conventional usability testing, because it enables evaluators to be "separated in space and/or time from users" [7]. The first methods were presented about 15 years ago, and some empirical studies were conducted [13]. More recently, it has been demonstrated that remote synchronous usability testing performs similarly to conventional user-based testing in a laboratory [25] while reducing main logistical challenges, most notably the need to get users to the lab [4, 12]. Unfortunately, the synchronous approach does not resolve the classical challenge of traversing hours of video recordings to identify usability problems [19].

The asynchronous approach moves remote usability testing a step further by involving prospective users. They identify and describe usability problems while using the system that is being tested. The asynchronous approach has strong advantages; in particular it reduces the time spent by the evaluators considerably. Yet there are still challenges that must be resolved before it can produce results that are comparable to conventional and remote synchronous approaches [2, 5]. Here, we focus on two challenges.

First, most studies of remote asynchronous methods employ task assignments to ensure that the users experience certain aspects of the system [2, 5, 6, 7, 12]. Yet considerable knowledge of the usage domain is needed to define good task assignments [11, 16], and predefined tasks compromise validity, because users are forced into artificial usage situations [3]. Alternatively, users could work with their own authentic tasks while reporting usability problems.

Second, some type of instruction is needed to enable users to report usability problems. Training users in this complex task is challenging. Early studies of remote asynchronous testing involved extensive training with instructors and users physically present together [6, 7, 28]. Yet that solution contradicts the whole idea of remote testing. Alternatively, users could receive written instructions over the Internet [2].

This paper presents a comparative study with the aim of examining the effect of task assignments and instruction types on the number of identified problems and problem variability within a remote asynchronous usability test. In the following section, we provide the theoretical background for the study and formulate five hypotheses for the study. Then we present an overview of related work on remote asynchronous testing related to these hypotheses. This is followed by a description of the experimental method applied in the study. The next section presents the results of the study. Then we discuss the findings and compare them to related work. Finally we provide the conclusion.

CHI 2012, May 5-10, 2012, Austin, TX, USA.

BACKGROUND AND HYPOTHESES

This section discusses practical guidance and theories related to task assignments and instruction types. The discussion is summarized in a set of hypotheses.

Task Assignments

For many years one of the most disputed guidelines related to usability testing in practice is the number of users needed to achieve a satisfactory outcome [17]. Lindgaard and Chattratichart analysed usability reports written by nine professional usability teams that tested the same interface. Their findings show no significant correlation between the number of users and the number of severe problems identified. However, there was a significant correlation between the number of tasks and the number of problems identified where higher task coverage causes a higher number of problems to be identified [20]. This indicates that task coverage has more impact on the number of identified usability problems than the number of test users.

Hertzum and Jacobsen reviewed eleven research papers to examine whether the impact of the evaluator effect could be dismissed as mere chance. The results in reviewed studies indicate that a vague goal analysis, which includes design of task assignments, causes increased variability in the set of problems identified by evaluators [13]. This in turn leads to a low any-two agreement.

Based on these research results, we have defined the following hypotheses regarding task assignments:

H1: Conditions with higher task coverage reveal more usability problems than conditions with lower task coverage.

H2: Conditions with no predefined tasks introduce more variability in identified usability problems compared to conditions based on predefined tasks.

Instruction Types

When users become directly involved in usability testing, they need training in identifying usability problems. If the training requires the users to come to a physical meeting, like in [6, 7, 28], it defies the very purpose of remote usability testing. There is little research on which instructions to apply as no previous studies have focused explicitly on the effect of different types of instructions in a remote setting. It is more in line with the idea of remote testing, if the instructions are delivered over the Internet, and they must be clear without being burdensome as users are interested in getting their work done as quickly as possible [7, 15, 26]. In our study, we distinguish between deductive and inductive instructions [9].

Deductive instructions reflect a classical way of conveying information, e.g. in engineering and science [24]. The teacher presents a general rule (or definition) to be learned, after which the learners reason on observations or examples that fit within the rule. Thus learners are told up front exactly what they need to know, which makes it a straightforward and well-structured approach to teaching.

Inductive instructions cover several approaches, including problem-based learning and discovery learning, which share the same underlying principle [24]. Specific observations or examples are presented initially, and then learners infer the general rule [24]. The examples must be familiar to the learners in order to create the best possible conditions for them to assimilate the new knowledge within their existing knowledge structures [24]. It has been argued that inductive instructions motivate learners to a greater degree than traditional deductive instructions hereby making induction more effective with respect to learning outcome [9, 24]. However, inductive instructions may also cause students to infer the wrong rule or the rule may be too narrow in its application [29]. This is a consequence if the examples are too few, narrow and not concise enough [9].

Combined deductive and inductive instructions is the third option. Some learners are best stimulated by deductive instructions while others prefer induction [9], thus an obvious idea is to combine them to stimulate both types.

Based on these theoretical considerations, we have defined the following hypotheses regarding instruction types:

H3: A) Inductive instructions cause users to identify more problems than by deduction. B) Instructions based on a combination of deduction and induction cause users to identify more problems than those receiving pure inductive or deductive instructions.

H4: Inductive instructions introduce the bias that users identify problems of the same category as the examples.

H5: Instructions based on a combination of induction and deduction will be preferred over the individual types.

RELATED WORK

This section presents related empirical work on remote asynchronous methods. The discussion is structured by our focus on the use of task assignments and the instruction types employed to train users in identifying usability problems. Table 1 provides an overview of the literature.

	Task assignments	
Instruction type	**Tasks**	**No tasks**
Deductive		
Inductive	6, 7, 12	
Combined deductive and inductive	5	
Not explicit	2, 21, 28	1, 3
No training	30, 31	27

Table 1: Overview of related work categorized according to our focus on task assignments and instruction types.

Three papers, based on the same study, describe training of users with an inductive approach based on specific

examples. This was done through a video presentation and hands-on exercises where the users were physically present [6, 7, 12]. In relation to task assignment, users were given 6 tasks to solve while participating. The objective in this study was to develop a remote asynchronous method, known as user-reported critical incidents (UCI), and investigate its feasibility. The effectiveness of an example-based video and hands-on exercise is compared between two groups of users. The type of instruction did not reveal any differences with respect to the number of usability problems identified.

A single study based user training on a combination of deductive and inductive instructions by providing learners with a definition of what a usability problem is and then showing examples [5]. The experiment required the users to solve 9 tasks. Training was done through written instructions sent to the users over the Internet.

Three studies do not state explicitly which instruction type they employ for user training. All three experiments are based on users solving tasks with given systems. Two of these study the effectiveness of the UCI method compared to a conventional lab setting [2, 28]. In [2], users were given 9 tasks to solve. In [28], the total number of tasks given is not mentioned. The final paper in this category describes a comparative study of a remote asynchronous method and a laboratory based method [21]. In this case the focus is not the UCI method but rather on a more open-ended reporting format. The users solved 5 tasks.

Two studies are not based on task solving. They are not explicit on the instruction type used. One of these focuses on evaluating the feasibility of the UCI method by comparing this to laboratory testing [3]. The other paper in this category is a comparative study of a remote asynchronous method and laboratory and expert inspection methods [1]. That paper does not study the UCI method but focus on a more open-ended reporting format.

Two papers describe remote usability studies in which users did not receive training in problem identification. In [31], auto logging, used for a formative test, is compared to of a traditional lab test. Yet the number of tasks is not reported. In [30], the effect of letting users fill in closed or semi-closed questionnaires based on customized frameworks was studied. The users solved 17 tasks.

In the last study, the users did not receive any training in identifying usability problems nor did they solve any tasks while testing systems. The users filled in a closed questionnaire based on a customized framework [27].

The research reviewed above primarily comprises feasibility studies of the performance of remote asynchronous methods compared to user-based laboratory testing or expert inspections. Only one of the studies focuses on different instructions. They compare an example-based video and a set of hands-on exercises [6]. This is, however, outside our aim of comparing inductive, deductive and combined instructions, because both of their types of are inductive and users were physically present during the training sessions. There is more research on task assignments as 9 papers provide tasks for users to solve while 3 papers do not. However, none of them compare conditions with and without task assignments.

METHOD

This section describes the method of our experiment. It included six remote asynchronous conditions. A conventional laboratory-based test [25] was used as a benchmark as this is common practice in usability testing research. The seven conditions are described below and summarized in Table 2.

	Tasks	**No tasks**
Deductive	DT (n=8)	DN (n=8)
Inductive	IT (n=7)	IN (n=6)
Deductive & Inductive	DIT (n=8)	DIN (n=6)
Laboratory testing	LAB (n=10)	

Table 2: Overview of the seven conditions (n=number of participants).

Task Assignments

The use of task assignments was one independent variable in the experiment.

Tasks. The test participants in three of the remote conditions received nine predefined task assignments to solve while using the system. These conditions are denoted as DT, IT and DIT (T for 'Task') and appear in the left hand column of Table 2. The tasks were derived from an interview with a manager and a secretary from the application domain. Each participant received a list with the nine tasks appearing in randomized order. The participants were asked to solve these tasks within four weeks and report the usability problems they had experienced.

No tasks. In the other three remote conditions, no predefined tasks were given. These conditions are denoted as DN, IN and DIN (N for 'No task') and appear in the right hand column of Table 2. The participants were asked to report the usability problems they experienced during their daily use, i.e. when using the system for their own purposes, and do it within a timeframe of four weeks.

Instruction Types

The instruction type was the other independent variable in the experiment. In all remote conditions, the participants received written instructions for training them in identification of usability problems. Above, it was emphasized that instructions provided remotely must be simple [7, 15, 26]. Therefore, the instructions were limited to a half page.

Deductive instructions. The test participants in two of the remote conditions received deductive instructions. These conditions are denoted as DT and DN (D for 'Deductive')

and appear in the first row of Table 2. For these conditions, we devised a purely deductive instruction by providing the general rule in the form of a conceptual definition of what a usability problem is. The definition was an inverted definition of a usable system as defined by Molich, cf. [22], combined with 1-2 lines of clarification for each of the following elements: "Not useful", "Difficult to learn", "Difficult to remember", "Ineffective to use" and "Unsatisfying to use".

Inductive instructions. The test participants in two of the remote conditions received inductive instructions. These conditions are denoted as IT and IN (I for 'Inductive') and appear in the second row of Table 2. For these conditions, we provided examples of usability problems but no definition. The participants were expected to use the examples to derive the general rule of what a usability problem is. We gave two examples of usability problems related to consistency and affordance; cf. [23] for description of these categories of usability problems. We chose to provide one example from Facebook and one from MS Word as the participants were familiar with these systems. According to the theory of inductive learning, these examples could fit into existing mental structures.

Combined deductive and inductive instructions. The test participants in two of the remote conditions received a combination of the deductive and inductive instructions. These conditions are denoted as DIT and DIN (DI for 'Deductive and inductive') and appear in the third row of Table 2. For these conditions, we combined the deductive and inductive instructions to form one page of instructions.

System

The system that was tested in the experiment was a website for a school in a university that offers a range of educations in information technology (www.sict.aau.dk). The website provides information about study regulations, educations, exams, study board members, contact information, campus maps etc. The majority of functionality on the core website enables students to retrieve information about educations, organization etc., and that is the part we tested. There are links to other websites with more interactive functions that we did not test, e.g. signing up for exams.

Participants

The participants in the experiment were students from various graduate and under-graduate educations in computer science, software engineering, informatics and other ICT related areas. Recruitment was done via an online screening survey sent to all students in the school. We promised the students that their participation in the experiment would only take a limited amount of time (a total of 1 hour). Through the survey, we collected demographic information such as education, age, sex and experience in using the system to be tested.

73 students responded to the survey. To limit the effort for each student, we decided to conduct a between-subjects experiment. They were evenly distributed over all conditions based on their demographic profiles.

53 participants completed the experiment. Thus the dropout rate was 24%, which is the same drop-off rate as reported in [30]. Table 2 shows the distribution of the 53 participants on the seven conditions. All participants received a gift.

Procedure for the Remote Conditions

For the remote conditions, we decided to use the UCI method, because it had demonstrated the best performance among existing remote asynchronous methods [5].

The participants received instructions as described above. Reminders were sent once every week, and participants who had not reported any problems were sent a reminder before the end of the period. This was done because a previous study has described problems with participants not reporting, which was attributed to the researchers failing to send reminders during the experiment [18].

Setting

The participants were not required to work with the system in a specific setting. They worked at home or at the university using their own computers.

Procedure and Data Collection

In accordance with the UCI method, participants were instructed to report any usability problem they found on the website as soon as they discovered it. This was done using a web-based report form that was programmed using PHP, JavaScript and a MySql database. The participants received a unique login and a link to the online report form. The participants in the DT, IT and DIT conditions also received their list with task assignments.

When the participants logged in, they were presented with the instructions pertaining to their specific condition. When they had finished reading, they pressed the "Start" button and were redirected to the report form. The form was similar to that used in other UCI experiments [5, 6, 7, 12].

The following points had to be answered using this form: Task (only for the IT, DT and DIT conditions), title of the webpage in which the problem occurred, intention, expectation, problem description, problem work-around and problem severity. At the bottom of the form, there was a submit button. When it was pressed, the data were saved in the MySQL database and the form was reset, ready for a new entry. The form was running in a separate browser window, so the participants toggled between the windows each time they encountered a problem.

Procedure for the Laboratory Condition

For the lab condition, we conducted a conventional user-based testing, cf. [25]. It was not conducted by the authors of this paper.

Setting

The test was conducted in a state-of-the art usability laboratory. In the test room, the test participant sat in front of the computer and next to her/him was a test monitor whose primary task was to ensure that the test participant was thinking aloud.

Procedure and Data Collection

The test participants were introduced to the test sequence and the concept of thinking aloud by the test monitor. We scheduled one hour per participant. The participants had to solve the nine tasks that were used in the DT, IT and DIT conditions while thinking aloud. Each participant received the nine tasks in randomized order. A video of the computer's desktop and a small picture of the participants' face were recorded.

Data Analysis

The data analysis was conducted by the two authors of this paper and four external evaluators who did not otherwise take part in the experiment. All six analysed the data from the remote conditions and three of them also analysed the video material from the lab test.

All data was collected before conducting the analysis. It consisted of 53 data sets (10 videos and 43 problem reports). Each data set was given a unique identifier, and a random list was generated for each evaluator, defining the order of analysis for all data sets. This was done to reduce an ordering bias. Each evaluator analysed all the data sets alone, one at a time. For the lab condition, the videos were thoroughly analysed trough a classical video analysis.

The data sets from the six remote conditions were analysed by reading one problem report at a time. By using only the information available in the users' problem description, it was transformed into a usability problem description. If necessary, the website was checked to get a better understanding of the problem.

Problem reports from the remote conditions were validated by considering the comprehensiveness of the wording, i.e. that the problem was formulated in such a way that we could understand the problem and locate it in the user interface. This is similar to the validation procedure described in [3]. If a description could not be translated into a meaningful problem in short time or the problem could not be identified using the website, the problem was not included in the problem list. During the analysis, the evaluators also rated the severity of the problems by using the categories of critical, serious and cosmetic.

Each evaluator created a problem list containing problems from all data sets. These lists were merged to form a joint problem list for all evaluators. In case of disagreement, it was negotiated by referring to the website and the original data until agreement was reached. Severity ratings in the joined lists were made by using the most serious rating. The resulting problem list included a detailed description of each usability problem.

The evaluators identified 42 usability problems in total (15 critical, 15 serious and 12 cosmetic). Thirty-eight instances from the UCI problem reports could not be turned into problems as they were impossible to understand. Of these, 6 were from the DT condition, 9 from DN, 10 from IT, 5 from IN, 3 from DIT and 5 from DIN. A Kappa inter-rater reliability measure shows a fair agreement ($0.4 <= p <= 0.56$) between evaluators on these 38 instances [10]. This agreement may seem low, but for a complex task like usability problem identification, it is actually at the better end of the scale [13].

RESULTS

In this section, we present our findings on the effect of using task assignments and different instruction types on the outcome of a remote asynchronous usability testing.

Effect of Task Assignments

The left hand box plots in Figure 1 provides an overview of the number of tasks solved per participant in pooled non-task based and task based conditions. For the task-based conditions we asked all participants to solve the 9 tasks provided. However, in case of non-task based conditions, we have no information about the total number of tasks that each participant tried to solve. An approximation can be derived through the data given in participants' problem descriptions, but this does not include information on whether a task was attempted solved, but no report of problems was generated. To make a fair comparison between the task based and non-task based conditions, we base all numbers of solved tasks only on the data given in participants' problem descriptions.

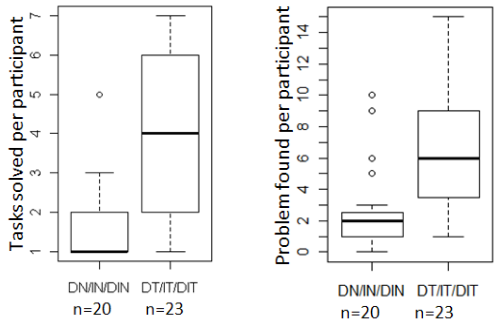


Figure 1: Left - Number of tasks attempted solved per participant (reported in problem descriptions). Right - Number of problems found per participant. The circular points indicate outliers.

The participants in the task-based conditions (DT/IT/DIT) attempted to solve a mean of 3.8 tasks (SD=1.96, n=23). In the non-task based (DN/IN/DIN) each participant attempted to solve a mean of 1.5 tasks (SD=1.04, n=19) when the outlier is removed (the circular point in Figure 1). A two-

sample t-test reveals a highly significant difference (t=4.97, df=38, p<0.001) between these conditions.

The right hand box plots in Figure 1 provides an overview of the number of problems identified per participant and shows that subjects in the task based conditions (DT/IT/DIT) generally identify more problems than in the non-task based (DN/IN/DIN). By removing the four outliers, each participant in DN/IN/DIN on average finds fewer problems (μ=2.16, SD=2.48, n=16) than in the DT/IT/DIT conditions (μ=6.67, SD=3.82, n=23). A two-sample t-test indicates that this difference is highly significant (t=-4.67, df=39.644, p<0.001). Looking at the number of problems found per task reveals that users in the remote task based conditions identifies a mean of 1.83 problems per task (SD=0.62) and that users in the non-task based settings on average finds 1.25 problems per task (SD=0.94). In this respect a two-sample t-test reveals no significant difference on this matter (t=-2.33, df=29.364, p>0.02).

	Critical	Serious	Cosmetic	Total
DN/IN/DIN n=20	4	6	3	**13 (31)**
DT/IT/DIT n=23	13	10	6	**29 (69)**
LAB n=10	12	13	11	**36 (86)**
Total	**15**	**15**	**12**	**42 (100)**

Table 3: Total number of problems found by pooling remote conditions with identical use of task assignments (n=number of participants). Numbers in parenthesis indicate percentage of the total.

Table 3 provides an overview of the total number of problems identified within pooled task based, non-task based conditions and LAB. By giving remote participants task assignments they are able to identify a total of 29 problems (69%) of which 13 are critical, 10 serious and 6 cosmetic. By not providing any predefined tasks users report a total of 13 problems (31%) when the four outliers shown in the right hand box plot on Figure 1 are removed. Of these problems, 4 are critical, 6 serious and 3 cosmetic. In the LAB condition 36 problems were identified (86%) of which 12 are critical, 13 serious and 11 cosmetic.

DT (n=21)	DN (n=36)	IT (n=28)	IN (n=6)	DIT (n=45)	DIN (n=15)	LAB (n=3)
0.24 (0.17)	0.06 (0.2)	0.33 (0.14)	0.02 (0.06)	0.27 (0.19)	0.05 (0.12)	0.54 (0.1)

Table 4: Mean any-two agreement. Parentheses indicate standard deviations (n=number of unique pairs of users/evaluators).

A Fishers exact test reveals highly significant differences in the total number of problems identified between remote task based and non-task based conditions (p<0.01). This also applies when comparing LAB and non-task based. However, we see no significant differences between remote task based conditions and LAB (p>0.1). This shows that by solving more tasks, participants identify significantly more problems. Yet, the number of problems found per task is similar performance between users in task based and non-task based conditions.

Problem Agreement

Table 4 shows the mean any-two agreement between users reporting problems in each of the remote conditions and the three evaluators who analysed video data in the LAB condition. The highest any-two agreement is seen between the three evaluators in the conventional LAB condition (μ=0.54, SD=0.1). The remote conditions have lower agreement; user agreement in all the task based conditions is higher (μ=[0.24;0.27;0.33], SD=[0.14;0.17;0.19]) than between users in non-task based (μ=[0.02;0.05;0.06], SD=[0.06; 0.12;0.2]). A one-way ANOVA test of all means shows significant differences between one or more of the conditions (df-resid=147, F=13.09, p<0.001). A Tukey's pair-wise comparison test reveals significant difference between all task based and non-task based conditions where 0.001<p<0.03. There is no significant difference between the LAB and remote task based conditions. Thus, the non-task based conditions cause users to have significantly less overlap in identified problems than in the task based and LAB conditions.

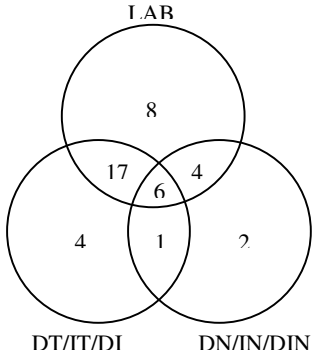


Figure 2. Venn diagram illustrating problem agreement between LAB and remote conditions pooled according to identical use of task assignments.

The agreement on the total number of usability problems identified across all conditions is illustrated in Figure 2. The pooled remote conditions identify 34 problems (the two bottom circles) with an agreement on 7 (21%). The remote task based conditions reveal 21 problems (61%) not found by the non-task based, while the latter uniquely identify 6 problems (18%). A Kappa inter-rater measure shows a poor agreement between remote task-based and non-task based conditions (p<0.4) [10].

The non-task based conditions and LAB together (the bottom right and top circle) identify a total of 38 problems. In this case 10 problems are agreed upon (26%), the non-

task based conditions uniquely find 3 problems (8%) and LAB uniquely finds 25 problems (66%). This corresponds to a poor Kappa agreement (p<0.4).

The LAB and remote task based conditions together (the bottom left and top circle) identify a total of 40 problems with an agreement on 23 (58%). The remote task based conditions reveal 5 problems not found by LAB (12%), while LAB identifies uniquely identifies 12 problems (30%). This corresponds to a good Kappa agreement (0.57<=p<=0.75). Thus, findings reveal a good agreement between task-based settings and a poor agreement between task-based and non-task based.

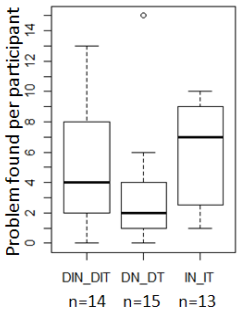


Figure 3: Number of problems found per participant in conditions pooled according to identical instruction type. The circular point indicates an outlier.

Effect of Instruction Types

The box plot in Figure 3 shows the number of problems identified per participant when pooling conditions with identical instruction types. Users who received deductive instructions identified fewer problems on average (μ=3.13, SD=3.67, n=16) than users receiving inductive instructions (μ=6.01, SD=3.26, n=13) or combined instructions (μ=5.31, SD=4.32, n=14).

	Critical	Serious	Cosmetic	Total
DIN/DIT (n=14)	10	9	6	**25 (59)**
DN/DT (n=15)	9	8	0	**17 (40)**
IN/IT (n=13)	13	10	6	**29 (69)**
LAB (n=10)	12	13	11	**36 (86)**

Table 5: Total number of problems found by pooling remote conditions with identical instruction types (n=number of users). Numbers in parenthesis indicate percentage of the total.

When removing the outlier in the DN/DT condition (the circular point in Figure 3), a one-way ANOVA test of all user means reveals significant differences between one or more of the pooled conditions (df-resid=40, F=4.968, p<0.02). A Tukey's pair-wise comparison test indicates significant difference between the DN/DT and DIN/DIT conditions (p<0.05) as well as DN/DT and IN/IT (p<.0.02). The difference between the DIN/DIT and IN/IT conditions is not significant (p>0.1). Thus, the inductive or combined instruction types cause users to identify significantly more problems than instructions based on deduction.

Table 5 shows the total number of problems identified in remote conditions pooled by identical instruction types and the LAB condition, when removing the outlier shown in Figure 3. For the remote settings, the inductive instructions reveal most problems (29) compared to deductive or combined instructions which uncover 17 and 25 problems respectively.

A Fishers exact test reveals a significant difference in the total number of identified problems between the IN/IT and DN/DT conditions (df=1, p<0.02). There is no significant difference between IN/IT and DIN/DIT (df=1, p>0.1). Furthermore we see a significant difference between the LAB-DN/DT and LAB-DIN/DIT conditions (df=1, p<0.01). Thus conditions in which users received inductive instructions identified significantly more problems in total than conditions where users were given deductive instructions.

Problem Types

We examined the types of problems, cf. [23], identified in all conditions to uncover whether users who received inductive instructions were biased towards identifying problems of the same type as in the examples provided in the inductive instruction (affordance and consistency). This revealed no significant differences.

Ratings of Instruction type

Users in the remote conditions gave satisfaction ratings of the instructions on a 5 point Likert scale (1=lowest satisfaction, 5=highest satisfaction). Table 6 shows the median ratings pooled by instruction types.

DN/DT (n=16)	IN/IT (n=13)	DIN/DIT (n=14)
4	5	4

Table 6: Median satisfaction ratings given on instruction types in remote conditions (n=number of participants).

A non-parametric Kruskal Walis test shows a significant difference between ratings for the DN/DT and IN/IT conditions (χ^2 = 4.4, df-con= 1, p<0.04) where induction is rated higher (median=5) than deduction (median=4). There is no significant difference between the DN/DT and DIN/DIT conditions (χ^2=0.32, df-con = 1, p>0.5) and IN/IT and DIN/DIT (χ^2=1.99, df-con=1, p>0.1). Thus, participants rate instructions based on induction higher than deduction or the combination of the two.

DICUSSION

In this section we discuss our findings in relation to the five hypotheses and related work.

H1: Tasks Solved and Problems Found

> **H1:** Conditions with higher task coverage reveal more usability problems than conditions with lower task coverage.

We found that task assignments improve the outcome of remote asynchronous usability testing. Thus we accept H1. Participants in the non-task based conditions on average solved significantly fewer tasks and in turn identified significantly fewer usability problems than participants in the task based conditions. This has also been found in a conventional lab setting [20]. It should, however, be noted that the number of problems found per task is similar for the task based and non-task based conditions.

We have not found studies of remote asynchronous testing that compare users solving predefined tasks with users working on their own problems. However, there are studies of either of these options. Three studies used no predefined tasks when comparing remote asynchronous methods to conventional lab or inspection methods. Two of these report that users applying remote methods identified between 67% and 73% of all problems [1, 27]- In the third, the users found more problems than the lab condition [3]. In our study we found that users in the pooled remote non-task based conditions found 31% of all problems, which is lower than the three studies. Unfortunately, none of these studies specify how many tasks the users solved. We found that participants solving more tasks identify more problems. If the users in the three studies solved more tasks than our users, that could explain the difference.

Nine other studies of remote asynchronous testing have used predefined tasks. In four of these, users applying remote asynchronous testing found between 52% - 68% of all problems [6, 7, 12, 30]. This is comparable to our findings, where participants in the pooled task based conditions identified 69% of all problems. Five other studies report lower numbers as their users in remote task based conditions uncovered between 21%-47% of all problems [2, 5, 21, 28, 31]. Four of these latter studies are either not explicit on the instruction type applied for training users or have not provided any. This may explain the difference.

H2: Task Assignments and Problem Variability

> **H2:** Conditions with no predefined tasks introduce more variability in identified usability problems compared to conditions based on predefined tasks.

We found that lack of predefined task assignments increased the variation among the usability problems identified. Our results indicate that participants in remote non-task based conditions have a significantly lower any-two agreement compared to participants in task based conditions and evaluators in the LAB condition. Thus we accept H2. This is consistent with a study concluding that a vague goal analysis causes an increased variability in the number of usability problems identified [13].

We found a good agreement between task-based settings and a poor agreement between task-based and non-task based. Thus users who did not receive predefined tasks had significantly less overlap in identified problems. All the participants who did not receive predefined tasks attempted to solve a total of 14 tasks. Four of these were similar to four of the 9 predefined tasks given to participants in task based conditions. Thus across all users, 10 tasks were uniquely solved in non-task based conditions, which gave a focus on other areas of the system, i.e. users in these conditions saw different parts of the website. This demonstrates that authentic system use fit well with exploratory tests where specific goals are missing. On the other hand, if there are specific areas of interest in an interface, users should be given predefined tasks to keep them within these limits.

We have found five studies that report problem agreement with remote asynchronous testing. Two of these report an agreement of 20% and 31% between remote task based and lab conditions [21, 30]. A third study found an agreement of 51% between a remote non-task based condition and expert inspection [27]. These results do not correspond to ours, as we have a higher agreement between remote task based and lab conditions (58%) and lower for of non-task based (8%). These differences may be caused by variations in instruction types [21] and lack of training [30, 27]. There is a higher correlation with related work for unique problems. Three studies report that, by merging problems found via task based remote conditions and lab, the remote conditions uniquely identify between 2% - 26% of all problems [2, 5, 30]. We find similar results as, compared to the lab condition, the remote task based conditions uniquely identified 21% of the problems.

H3 – Effectiveness of Inductive Instructions

> **H3:** A) Inductive instructions cause users to identify more problems than by deduction. B) Instructions based on a combination of deduction and induction cause users to identify more problems than those receiving pure inductive or deductive instructions.

By pooling the remote conditions with identical instruction types we found that each participant on average identified significantly more problems when given inductive instructions compared to those who were given deductive. Thus we accept H3(A).

The combination of deductive and inductive instructions also caused each participant to uncover significantly more problems than those who received deductive instructions only, which support H3(B). However, there was a tendency that the users receiving inductive instructions identified more problems than those who received the combination, but this difference was not significant. The latter finding ultimately causes us to reject H3(B).

We have found four papers that explicitly describe the type of instruction given to users. Three of these are based on the

same study and compare the effectiveness of two types of inductive instructions conveyed in physical presence [6, 7, 12]. The first of these is an example-based video. The second is a hands-on exercise. These two instructions are compared between two groups of users. The results did not reveal any differences in the number of problems identified. The users in the remote conditions identified 68% of the usability problems found in a conventional video-based analysis. This is similar to our findings where participants receiving inductive instructions, identified 69% of all problems. Another study used a combination of deductive and inductive instructions by presenting a definition of a usability problem and some examples [5]. The users trained this way were able to find between 21% and 47% of all problems, depending on the remote method applied. Our result for the similar condition was 59% of all problems.

H4 –Bias towards Instruction type

> **H4:** Inductive instructions introduce the bias that users identify problems of the same category as the examples.

We did not find that inductive instructions introduced a bias by causing users to infer a wrong rule or a rule too narrow in its application. Our inductive instruction provided two examples; one of an affordance problem and one of a consistency problem. Yet our results showed no significant differences that reflected a bias towards those two types of problems. Thus we reject H4.

We have found a single paper that describes the types of problems identified by users in a remote asynchronous condition [1]. However, we do not know the instruction type applied and, therefore, we do not know whether the problems identified were the same type as eventual examples.

H5 – Subjective Preferences of Instruction type

> **H5:** Instructions based on a combination of induction and deduction will be preferred over the individual types.

We did not find that the combination of the deductive and inductive instruction types was preferred by the users. We measured the subjective satisfaction with the instruction types across the users in the remote conditions. We did not find significant differences between instructions based on a combination and instructions exclusively based on one of the types. Thus we reject H5.

This result contradict advice in the literature which states that Instructions should be based on a combination of deduction and induction as this stimulates learners preferring either type [9]. An explanation why participants did not rate the combined instructions highest may be that such instructions results in more text which causes training to be more burdensome and time-consuming, which should be avoided in remote usability testing [7, 15, 26]. None of the studies in related work present findings on user ratings of instructions, hereby making a comparison to these impossible. It is also interesting that the users who received inductive instructions identified more problems than those receiving deductive or a combination thereof.

CONCLUSION

An increasing body of research demonstrates that remote asynchronous usability testing has promising benefits. However, there are still aspects that need to be developed. In this paper, we have presented a comparative empirical study of the effect of task assignments and instruction types on the result of a remote asynchronous usability test. The study joins a trend where mere method comparisons are replaced with practice-oriented studies of the effects of variations in method use.

Our findings show that users receiving predefined tasks solved significantly more tasks, identified significantly more usability problems and had a significantly higher level of problem agreement than those working on their own authentic tasks. Not providing predefined tasks caused the users to identify more varied sets of problems than when predefined tasks were provided. Finally, users who were instructed by means of inductive examples of usability problems identified significantly more usability problems than users who were given a deductive conceptual definition, and the satisfaction rating for the inductive instruction was significantly higher.

The results are limited by the type of website tested, which is mainly used for medium-regular information retrieval. Also, the sheer size of the website implies that different users may have experienced various parts that appear differently, this is particularly likely with the non-task users. Moreover, other types of users and systems may reveal different results, for example systems applied more frequently by expert users to solve more complex tasks.

In the future it would be relevant to conduct similar experiments based on other types of systems and users. It would also be interesting to resolve the basic challenge of collecting more information about user activity in a remote asynchronous test.

ACKNOWLEDGMENTS

The research behind this paper was partly financed by the Danish Research Councils (grant number 09-065143). We are grateful to the students who participated, to the school that facilitated the test and the anonymous reviewers.

REFERENCES

1. Äijö, R. and Mantere, J. Are Non-Expert Usability Evaluations Valuable? http://www.hft.org/HFT01/paper01/acceptance/2_01.pdf
2. Andreasen, M. S., Nielsen, H. V., Schrøder, S. O. and Stage, J. What happened to remote usability testing? An empirical study of three methods. In *proc. CHI 2007*, ACM Press (2007), 1405-1414.

3. Bosenick, T., Kehr, S., Kühn, M. and Nufer, S. Remote usability tests: an extension of the usability toolbox for online-shops. In *Proc. UAHCI 2007*, Springer-Verlag (2007), 392-398.
4. Brush, A. B., Ames, M. and Davis, J. A comparison of synchronous remote and local usability studies for an expert interface. In *proc. CHI 2004*, ACM Press (2004), 1179-1182.
5. Bruun, A., Gull, P., Hofmeister, L. and Stage, J. Let Your Users Do the Testing: A Comparison of Three Asynchronous Usability Testing Methods. In *proc. CHI 2009*, ACM Press (2009), 1619-1628.
6. Castillo, J. C. *The User-Reported Critical Incident Method for Remote Usability Evaluation*. Master thesis, Virginia Polytechnic Institute and State University (1997).
7. Castillo, J. C., Hartson, H. R. and Hix, D. Remote usability evaluation: Can users report their own critical incidents? In *proc. CHI 1998*, ACM Press (1998), 253-254.
8. Felder, R.M. Reaching the second tier: Learning and teaching styles in college science education. *College Science Teaching 23*, 5 (1993), 286-290.
9. Felder, R.M. and Silverman, L.K. Learning and Teaching Styles in Engineering Education. *Engineering. Education 78*, 7 (1988), 674-681.
10. Fleiss, J.L. *Statistical methods for rates and proportions (2nd ed.)*. John Wiley & Sons, New York, 1981.
11. Følstad, A. and Hornbæk, K. Work-domain knowledge in usability evaluation: Experiences with Cooperative Usability Testing. *Journal of Systems and Software 83*, 11, (2010), 2019-2030.
12. Hartson, H. R. and Castillo, J. C. Remote evaluation for post-deployment usability improvement. In *proc. AVI 1998*, 22-29.
13. Hartson, H. R., Castillo, J. C., Kelso, J. and Neale, W. C. Remote evaluation: The network as an extension of the usability laboratory. *Proceedings of CHI 1996*, ACM Press (1996), 228-235.
14. Hertzum, M. and Jacobsen, N.E. The Evaluator Effect: A Chilling Fact about Usability Evaluation Methods. *Human Computer Interaction 15*, 1 (2003), 1336-1340.
15. Hilbert, D.M. and Redmiles, D.F. Separating the Wheat from the Chaff in Internet-Mediated User Feedback Expectation-Driven Event Monitoring. *ACM SIGGROUP Bulletin 20*, 1, (1999), 35-40.
16. Hornbæk, K. and Frøkjær, E. Making Use of Business Goals in Usability Evaluation: An Experiment with Novice Evaluators. In *proc. CHI 2008*, ACM Press (2008), 903-911.
17. Hwang, W. and Salvendy, G. Number of people required for usability evaluation: the 10±2 rule. *Commun. ACM* 53, 5 (May 2010), 130-133.
18. Kjaer, A., Madsen, K.H. and Petersen, M.G.. Methodological Challenges in the Study of Technology Use at Home. In *proc. HOIT 2000*, Kluwer Academic Publishers (2000), 45–60.
19. Kjeldskov, J., Skov, M. B. & Stage, J. Instant Data Analysis: Evaluating Usability in a Day. In *proc. NordiCHI 2004*, ACM Press (2004), 233-240.
20. Lindgaard, G. and Chattratichart, J. Usability Testing: What Have We Overlooked? In *proc. CHI 2007*, ACM Press (2007), 1415-1424.
21. Marsh, S. L., Dykes, J. and Attilakou, F. Evaluating a geovisualization prototype with two approaches: remote instructional vs. face-to-face exploratory. In *proc. Information Visualization 2006*, IEEE (2006), 310-315.
22. Molich, R. *Usable Web Design*. Nyt Teknisk Forlag, Odense, Denmark, 2007.
23. Nielsen, C. M., Overgaard, M., Pedersen, M.B., Stage, J. and Stenild, S. It's Worth the Hassle! The Added Value of Evaluating the Usability of Mobile Systems in the Field. In *proc. NordiCHI 2006*, ACM Press (2006), 272-280.
24. Prince, M.J. and Felder, R.M. Inductive teaching and learning methods: Definitions, comparisons, and research bases. *Engineering Education 95* (2006), 123-138.
25. Rubin, J. and Chisnell, D. *Handbook of Usability Testing: How to Plan, Design, and Conduct Effective Tests*. Wiley Publishing, Indianapolis, USA, 2008.
26. Scholtz, J. A case study: developing a remote, rapid and automated usability testing methodology for on-line books. In *proc. HICSS 1999*, IEEE (1999).
27. Ssemugabi, S. and Villiers, R.D. A comparative study of two usability evaluation methods using a web-based e-learning application. In *proc. SAICSIT 2007*, ACM Press (2007), 132-142.
28. Thompson, J. A. *Investigating the Effectiveness of Applying the Critical Incident Technique to Remote Usability Evaluation*. Master thesis, Virginia Polytechnic Institute and State University, 1999.
29. Thornbury, S. *How to teach grammar*. Pearson Education Ltd, Harlow, Essex, England, 1999.
30. Tullis, T., Fleischman, S., McNulty,M., Cianchette, C. and Bergel, M. An empirical comparison of lab and remote usability testing of web sites. http://home.comcast.net/~tomtullis/publications/RemoteVsLab.pdf
31. Waterson, S., Landay, J. A. and Matthews, T. In the lab and out in the wild: remote web usability testing for mobile devices. In *proc. CHI 2002*, ACM Press (2002), 796-797.
32. Winckler, M. A. A., Freitas, C. M. D. S. and de Lima, J.V.Remote usability testing: a case study. In *proc. OzCHI 1999*, CHISIG (1999).

8 Appendix B – Equations for Quality Criteria

Thoroughness

(1) $$\frac{|P \cap LAB|}{|LAB|}$$

Validity

(2) $$\frac{\left|P_{UEM_i} \cap LAB\right|}{\left|P_{UEM_i}\right|}$$

Reliability

(3) $$Avg.of\ \frac{|P_i \cap P_j|}{|P_i \cup P_j|}\ for\ all\ pairs$$

Downstream Utility

(4) $$CIR = \frac{No.of\ problems\ comitted\ to\ fix}{No.of\ identified\ problems}$$

(5) $$CDIR = \frac{No.of\ problems\ fixed}{No.of\ identified\ problems}$$

Cost Effectiveness

(6) $$\frac{Total\ time\ spent}{No.of\ problems\ identified}$$

9 Appendix C – References in Literature Review

References cited in contribution 1.

1. Aikio, K.-P., 2007. Exporting Usability Knowledge into a Small-Sized Software Development Organization – A Pattern Approach. In Smith, M.J. and Salvendy, G. (Eds), *HCI* (pp. 3-11). Springer-Verlag, Berlin, Germany.
2. Ardito, C., Costabile, M.F., De Angeli, A. and Lanzilotti, R., 2006. Systematic Evaluation of e-Learning Systems: An Experimental Validation. In *Proceedings of the 4th Nordic conference on Human-computer interaction*. ACM, New York, NY, USA.
3. Bailey, G., 1993. Iterative methodology and designer training in human-computer interface design. In *Proceedings of the INTERACT '93 and CHI '93 conference on Human factors in computing systems*. ACM, New York, NY, USA.
4. Barber, J., Bhatta, S., Goel, A., Jacobson, M., Pearce, M., Penberthy, L., Shankar, M., Simpson, R. and Stroulia, E., 1992. AskJef: Integration of Case-Based and Multimedia Technologies for Interface Design Support. In *Proceedings of the Second International Conference on AI in Design*. Kluwer Academic.
5. Blandford, A. E., Shum, S. J. B. and Young, R. M., 1998. Training Software Engineers in a Novel Usability Evaluation Technique. In *International Journal of Human-Computer Studies, 49*(3), pp. 245-279. Academic Press Inc., Duluth, MN, USA.
6. Borchers, J., 2002. Teaching HCI design patterns: Experience from two university courses. *Workshop at CHI 2002 International Conference on Human Factors of Computing Systems*. ACM, Minneapolis, MI, USA.
7. Brandt, C., Cennano, K., Douglas, S., McGrath, M., Reimer, Y., and Vernon, M., 2008. (de)Coding the Studio Method to Teach the Design of Human-Computer Interaction. In *Proceedings of the 24th National Conference on the Beginning Design Student*. Georgia Institute of Technology.
8. Brown, B., Lundin, J., Rost, M., Lymer, G. and Holmquist, L.E., 2007. Seeing Ethnographically: Teaching ethnography as part of CSCW. In *Proceedings of the 10th European Conference on Computer-Supported Cooperative Work*. Springer, London, UK.
9. Brown, C. and Pastel, R., 2009. Combining Distinct Graduate and Undergraduate HCI Courses: An Experiential and Interactive Approach. In *Proceedings of the 40th ACM technical symposium on Computer science education*. ACM New York, NY, USA.
10. Calderòn, M.E., 2009. Teaching Human Computer Interaction: First Experiences. In *CLEI Electronic Journal, 12*(1).
11. Calogne, C.M., 2001. Designing for Web Site Usability. In *Proceedings of the seventh annual consortium for computing in small colleges central plains conference on The journal of computing in small colleges*. Consortium for Computing Sciences in Colleges, USA.
12. Cervone, H. F., 2005. Usability Training: An Overlooked Component in an On-Going Program of Web Assesment and Development. In *OCLS Systems and Services*. Emerald Group Publishing, Great Britain.
13. Chan, S. S., Wolfe, R. J. and Fang, X., 2002. Teaching HCI in IS/EC Curriculum. In *Proceedings of the Eighth Americas Conference on Information Systems*.
14. Chattratichart, J., 2007. A Theory-Based Approach to Designing Student Learning Context. In *CHI '07 extended abstracts on Human factors in computing systems*. ACM, New York, NY, USA.

15. Chevalier, A. and Ivory, M.Y., 2003. Can Novice Designers Apply Usability Criteria and Recommendations to make Web Sites Easier to Use? In *J. Jacko & C. Stephanidis (Eds.), Human-Computer Interaction, Theory and Practice (Part 1)*, pp. 58-62. Lawrence Erlbaum Associates, Mahwah, NJ, USA.
16. Chung, E.S., Hong, J.I., Lin, J., Prabaker, M.K., Landay, J.A. and Liu, A., 2004. Development and Evaluation of Emerging Design Patterns for Ubiquitous Computing. In *Proceedings of the 5th conference on Designing interactive systems*. ACM, New York, NY, USA.
17. Ciolfi, L. and Cooke, M., 2006. HCI for interaction designers: Communicating "the bigger picture". In *Proceedings of HCIEd2006-1 First Joint BCS/IFIP WG13.1/ICS/ EU CONVIVIO HCI Educators' Workshop*.
18. Cockburn, A. and Bell, T., 1998. Extending HCI in the Computer Science Curriculum. In *Proceedings of the 3rd Australasian conference on Computer science education*. ACM, New York, NY, USA.
19. Cowley, N.L.O. and Wesson, J. L., 2005. An Experiment to Measure the Usefulness of Patterns in the Interaction Design Process. In *Proceedings of the Tenth IFIP TC13 International Conference on Human-Computer Interaction (INTERACT 2005)*. Springer Verlag, Berlin, Germany.
20. Cross, J. H. et al., 2001. Computing Curricula 2001 – Computer Science. In *ACM Journal of Educational Ressources in Computing, 1*(3), Article #1. ACM, New York, NY, USA.
21. De Angeli, A., Matera, M., Constabile, M.F., Garzotto, F. and Paolini, P., 2003. On the Advantages of a Systematic Inspection for Evaluating Hypermedia Usability. In *International Journal of Human-Computer Interaction, 15*(3), pp. 315 - 335. Taylor & Francis.
22. Desurvire, H., Kondziela, J. and Atwood, M.E., 1993. What is Gained and Lost When Using Methods other than Empirical Testing. In *Proceedings of the conference on People and computers VII*. Cambridge University Press, New York, NY, USA.
23. Docherty, M., Sutton, P., Brereton, M. and Kaplan, S., 2001. An Innovative Design and Studio-based CS Degree. In *Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education*. ACM, New York, NY, USA.
24. Eierman, Michael A. and Iversen, Jakob, 2008. Teaching User Interface Design: An Important Addition to the Information Systems Curriculum. In *MWAIS 2008 Proceedings*.
25. Faiola, A. and Matei, S.A., 2009. Enhancing Human-Computer Interaction Design Education: Teaching Affordance Design for Emerging Mobile Devices. In *International Journal of Technology & Design Education*. Springer Science & Business Media B.V., Dordrecht, Netherlands.
26. Ferre, X., Juristo, N. and Moreno, A. M., 2003. Improving Software Engineering Practice with HCI Aspects. In *Software Engineering Research and Applications, First International Conference*. Springer-Verlag, Berlin, Germany.
27. Ferre, X., Juristo, N., Moreno, A.M., 2004. Deliverable D.5.2 Specification of the Software Process With Integrated Usability Techniques. STATUS Project. http://is.ls.fi.upm.es/status/results/STATUS_D5.2_v1.0.pdf
28. Ferre, X., Juristo, N., Moreno, A.M., 2005. Framework for Integrating Usability Practices into the Software Process. In *Product Focused Software Process Improvement, 6th International Conference*. Springer-Verlag, Berlin, Germany.
29. Fonseca, M.J., Jorge, J.A., Gomes, M.R., Goncalves, D. and Vala, M., 2009. Conceptual Design and Prototyping to Explore Creativity. In *P. Kotzé. W. Wong, J. Jorge, A. Dix and P. Silva (eds.), Creativity and HCI: From Experience to Design in Education*. Springer, Boston, MS, USA.
30. Frøkjær, E. and Lárusdòttir, M.K., 1999. Prediction of Usability: Comparing Method Combinations. In *Proceedings of the Information Resources Management Association International Conference (IRMA)*. Idea Group Publishing.

31. Golden, E., John, B.E. and Bass, L., 2005. The Value of a Usability-Supporting Architectural Pattern in Software Architecture Design: A Controlled Experiment. In *Proceedings of the 27th international conference on Software engineering*. ACM, New York, NY, USA.
32. Granger, M.J., 2007. Emphasizing the User in a Structured Human-Computer Interaction Course. In *Journal of Informatics Education Research (JIER), 9*(1), pp. 79 - 94.
33. Greenberg, S., 1996. Teaching Human-Computer Interaction to Programmers. In *Interactions, 3*(4), pp. 62-76. ACM, New York, NY, USA.
34. Greenberg, S., 2009. Embedding a Design Studio Course in a Conventional Computer Science Program. In *P. Kotzé. W. Wong, J. Jorge, A. Dix and P. Silva (eds.), Creativity and HCI: From Experience to Design in Education -- Selected Contributions from HCIEd 2007*. Springer, Boston, MS, USA.
35. Griffiths, R. N. and Pemberton, L., 2003. Teaching Usability Design through Pattern Language, (cited 2010-02-23); http://www.it.bton.ac.uk/staff/lp22/CHIpaper.html.
36. Häkli, A., 2005. *Introducing UCD in a Small-Size Software Development Organization*. Helsinki University of Technology, Helsinki, Finland.
37. Harrison, S. and Tatar, D., 2006. More Than a Method! In *Proceedings of the First Joint BCS/IFIP WG13.1/ICS/ EU CONVIVIO HCI Educators' Workshop*. Limerick, Ireland.
38. Hartfield, B., Winograd, T. and Bennett, J., 1992. Learning HCI Design: Mentoring Project Groups in a Course on Human-Computer Interaction. In *ACM SIGCSE Bulletin, 24*(1), pp. 246 - 251. ACM, New York, NY, USA.
39. Heinecke, A.M., Strauss, F., Beck, A., Dahm, M., Hamborg, K.-C. and Heers, R., 2008. What Every Software Developer Should Know about Human-Computer Interaction – A Curriculum for a Basic Module in HCI in Informatics Education. In *Proceedings of ACM - IFIP Informatics Education Europe III*. ACM, New York, NY, USA.
40. Hertzum, M. and Jacobsen, N.E., 1999. The Evaluator Effect during First-Time Use of the Cognitive Walkthrough Technique. In *Proceedings of HCI International*. L. Erlbaum Associates Inc., Hillsdale, NJ, USA.
41. Hewett, T., Baecker, R., Card, S., Carey, T., Gasen, J., Mantei, M., Perlman, G., Strong, G. And Verplank, W., 1992. *ACM SIGCHI Curricula for Human-Computer Interaction*. ACM, New York, NY, USA.
42. Holleis, P. and Schmidt A., 2008. MAKEIT: Integrate User Interaction Times in the Design Process of Mobile Applications. In *Proceedings of the 6th International Conference on Pervasive Computing*. Springer-Verlag , Berlin, Heidelberg.
43. Hornbæk, K. and Frøkjær, E., 2004a. Two Psychology-Based Usability Inspection Techniques Studied in a Diary Experiment. In *Proceedings of the third Nordic conference on Human-computer interaction*. ACM, New York, NY, USA.
44. Hornbæk, K. and Frøkjær, E., 2004b. Usability Inspection by Metaphors of Human Thinking Compared to Heuristic Evaluation. In *International Journal of Human-Computer Interaction, 17*(3), pp. 357-374. Lawrence Erlbaum Associates, Inc.
45. Howarth, J., Andre, T.S., and Hartson, R. A, 2007. A Structured Process for Transforming Usability Data into Usability Information. In *Journal of Usability Studies, 3*(1), pp. 7-23.
46. Iachello, G. and Abowd G., 2005. An Evaluation of the Comprehensibility and Usability of a Design Method for Ubiquitous Computing Applications. GVU Tech. rep. GIT-GVU-05-32, Georgia Institute of Technology, Atlanta, GA.
47. Jacobsen, N.E. and John, B.E., 2000. Two Case Studies in Using Cognitive Walkthroughs for Interface Evaluation. Technical report CMU-CS-00-132.

48. Jeffries, R., Miller, J.R., Wharton, C. and Uyeda, K.M., 1991. User Interface Evaluation in the Real World: A Comparison of Four Techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA.

49. John, B.E. and Mashyna, M.M., 1995. Evaluating a Multimedia Authoring Tool with Cognitive Walkthrough and Think-Aloud User Studies. In *Journal of the American Society of Information Science, 48*(9), pp. 1004–1022.

50. Jonathan, R. H., 2007. *Supporting novice usability practitioners with usability engineering tools*. Virginia Polytechnic Institute & State University Blacksburg, VA, USA.

51. Juristo N., Moreno, A. and Sanchez-Segura, M.-I., 2007a. Guidelines for enabling Usability Functionalities. In *IEEE Transactions on Software Engineering, 33*(11), pp. 744-758. IEEE Press, Piscataway, NJ, USA.

52. Juristo N., Moreno, A., Sanchez-Segura, M.-I. and Baranauskas, M. C. C., 2007b. A Glass Box Design: Making the Impact of Usability on Software Development Visible. In Baranauskas, C., *INTERACT* (pp. 541-554). Springer-Verlag, Berlin, Germany.

53. Karat, C.M., Campbell, R. and Fiegel, T., 1992. Comparison of Emprical Testing and Walkthrough Methods in User Interface Evaluation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA.

54. Karat, J. and Dayton, T., 1995. Practical Education for Improving Software Usability. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM Press/Addison-Wesley Publishing Co. New York, NY, USA.

55. Kay, J. and Kummerfeld, B., 1998. A problem-based interface design and programming course. In *Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education*. ACM, New York, NY, USA.

56. Khoo, B. and Preece, J., 1999. An Interactive Case Scenario for Teaching User Interface Design. In *Proceedings of the Fifth Americas Conference on Information Systems*.

57. Kiili, K., Multisilta, J., Suominen, M. and Ketamo, H., 2009. Learning Experiences on Mobile Social Media. In *Proceedings of the 17th International Conference on Computers in Education*. Asia-Pacific Society for Computers in Education, Hong-Kong.

58. Kilpatrick, C., 1996. Community Web Projects to Support Human-Computer Interaction Education. In *WebNet 96 Conference Proceedings*. Association for the Advancement of Computing in Education (AACE), Charlottesville, VA, USA.

59. Klemmer, S.R., Sinha, A.K., Chen, J., Landay, J.A., Aboobaker, N. and Wang, A., 2000. Suede: A Wizard of Oz Prototyping Tool for Speech User Interfaces. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*. ACM, New York, NY, USA.

60. Koppelman, H. and Van Dijk, B., 2006. Creating a Realistic Context for Team Projects in HCI. In *ACM SIGCSE Bulletin, 38*(3), pp. 58-62. ACM New York, NY, USA.

61. Kotzé, P., Renaud, K., Koukouletsos, K., Khazaei, B., and Dearden, A., 2006. Patterns, Anti-Patterns and Guidelines—Effective Aids to Teaching HCI Principles?. In *Proceedings of HCIEd2006*.

62. Koukouletsos, K., Khazaei, B., Dearden, A. and Ozcan, M., 2009. Teaching Usability Principles with Patterns and Guidelines. In *IFIP International Federation for Information Processing, 289*, pp. 159-174. Springer, Boston, MA, USA.

63. Koukouletsos, K., Khazaei, B., Dearden, A. and Tseles, D., 2007. An Empirical Comparison on the Evaluation of Web Sites with Usability Criteria. In *Proceedings of the International Scientific Conference e RA*.

64. Koutsabasis, P., Spyrou, T., Darzentas, J.S. and Darzentas J., 2007. On the Performance of Novice Evaluators in Usability Evaluations. In *Proceedings of the 11th Panhellenic Conference on Informatics*.

65. Lárusdóttir, M.K., 2006. Using rapid contextual design ar Reykjavik University. In *Proceedings of the First Joint BCS/IFIP WG13.1/ICS/ EU CONVIVIO HCI Educators' Workshop*. Limerick, Ireland.

66. Latzina, M. and Rummel, B., 2002. Collaboration-Based Usability Training for Developers. In *Mensch & Computer*. B. G. Teubner, Stuttgart, Germany.

67. Lauesen, S., 1997. Usability Engineering in Industrial Practice. In *Proceedings of the IFIP TC13 Interantional Conference on Human-Computer Interaction*. Chapman & Hall, Ltd. London, UK, UK.

68. Law, E. L.-C. and Hvannberg, E. T., 2004. Analysis of Strategies for Improving and Estimating the Effectiveness of Heuristic Evaluation. In *Proceedings of the third Nordic conference on Human-computer interaction*. ACM, New York, NY, USA.

69. Lazar, J., 2000. Teaching Web design though community service projects. In *Journal of Informatics Education and Research, 2*(2), pp. 71-75.

70. Leichtenstern, K. And André, E., 2009. The Assisted User-Centred Generation and Evaluation of Pervasive Interfaces. In *Proceedings of the European Conference on Ambient Intelligence*. Springer Verlag, Berlin, Heidelberg, Germany.

71. Lennon, M. and Bannon, L.J., 2006. Worksheets in Practice: Gathering Artefacts for Reflection in Interaction Design Education. In *Proceedings of the First Joint BCS/IFIP WG13.1/ICS/ EU CONVIVIO HCI Educators' Workshop*. Limerick, Ireland.

72. Leventhal, L. and Barnes, J., 2003. Two for One: Squeezing Human-Computer Interaction and Software Engineering into a Core Computer Science Course. In *Journal of Computer Science and Education, 13*(3), pp. 177 – 190.

73. Lewis, C., Polson, P., Wharton, C. and Rieman, J., 1990. Testing a Walkthrough Methodology for Theory-Based Design of Walk-Up-and-Use Interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA.

74. Lidtke, D.K. and Stokes, G., 1999. An Information Systems-Centric Curriculum, ISCC '99. In *Journal of Systems and Software, 49*(2-3), pp. 171-175. Elsevier Science Inc. New York, NY, USA.

75. Loréz, J., Abascal, J., Cañas, J.J., Aedo, I., Gea, M., Ortega, M., Ureña, L.A., Valero, P.M. and Velez, M., 2001. HCI Curricula in Spain. In *M. Ortega & J. Bravo (Eds.), Computers and Education*, pp. 227-234. Kluwer Academic Press, Netherlands.

76. Loréz, J., Granollers, T. and Aguiló, C., 2006. An Undergraduate Teaching Experience in HCI at the University of Lleida. In *The First Joint BCS/IFIPWG13.1/ICS/EU CONVIVIO HCI Educators' Workshop*. Limerick, University of Limerick.

77. Lowry, P.B., Dean, D.L., Roberts, T.L. and Marakas, G., 2009. Toward Building Self-Sustaining Groups in PCR-based Tasks through Implicit Coordination: The Case of Heuristic Evaluation. In *Journal of the Association for Information Systems, 10*(3), pp. 170 – 195.

78. Ludi, S., 2003. Undergraduate Software Engineering Curriculum Enhancement via Human-Computer Interaction. In *ICSE Workshop on SE-HCI*.

79. Manaris, B. and McCauley, R., 2004. Incorporating HCI into the Undergraduate Curriculum: Bloom's Taxonomy Meets the CC'01 Curricular Guidelines. In *34th ASEE/IEEE Frontiers in Education Conference*. Stipes.

80. Manaris, B., Wainer, M., Kirkpatrick, A.E., Stalvey, R.H., Shannon, C., Leventhal, L., Barnes, J., Wright, J., Schafer, J.B. and Sanders, D., 2007. Implementations of the CC'01 Human-Computer Interaction Guidelines Using Bloom's Taxonomy. In *Computer Science Education, 17*(1), pp. 21-57. Taylor & Francis Ltd., Philadelphia, PA, USA.

81. Matson, R., 2001. Re-Forming Information: A Case Study in Teaching Content Encapsulation. In *Proceedings of the 19th annual international conference on Computer documentation*. ACM, New York, NY, USA.

82. McCrickard, D.S. and Chewar, C.M. Creating a Broad, Interdiciplinary HCI Experience: A View of the HCI Certificate Program at Virginia Tech.

83. McCrickard, D.S., Chewar, C.M. and Sommervell, J., 2004. Design, Science and Engineering Topics? Teaching HCI with a Unified Method. In *ACM SIGCSE Bulletin, 36*(1), pp. 31-35.

84. Moroz-Lapin, K. and Ragaisis, S., 2007. Teaching HCI in SE Curriculum. In *Proceedings of the 4th WSEAS/IASME International Conference on Engineering Education*. WSEAS, Stevens Point, Wisconsin, USA.

85. Nielsen, J., 1992. Finding Usability Problems Through Heuristic Evaluation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA.

86. Nielsen, J., Bush, R.M., Dayton, T., Mond, N.E., Muller, M.J. and Root, R.W., 1992. Teaching Experienced Developers to Design Graphical User Interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA.

87. Nielsen, J. and Molich, R., 1989. Teaching User Interface Design Based on Usability Engineering. In *ACM SIGCHI Bulletin, 21*(1), pp. 45 - 48.ACM, New York, NY, USA.

88. Nielsen, J. and Molich, R., 1990. Heuristic Evaluation of User Interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA.

89. Obendorf, H., Schmolitzky, A. and Finck, M., 2006. XPnUE - Defining and Teaching a Fusion of EXtreme Programming and Usability Engineering. In *Proceedings of the First Joint BCS/IFIP WG13.1/ICS/ EU CONVIVIO HCI Educators' Workshop*. Limerick, Ireland.

90. Peslak, A., 2005. A Framework and Implementation of User Interface and Human-Computer Interaction Instruction. In *Journal of Information Technology Education, 4*, pp. 189-205. Informing Science Institute, Santa Rosa, CA, USA.

91. Plimmer, B., 2005. A Computer Science HCI Course. In *Proceedings of Human-Computer Interaction*.

92. Plimmer, B. and Amor, R., 2006. Peer Teaching Extends HCI Learning. In *ACM SIGCSE Bulletin, 38*(3), pp. 53-57. ACM, New York, NY, USA.

93. Polak-Wahl, J. A., 2004. Teaching HCI in Software Engineering. In *Frontiers in Education, 34*, pp. F1C - 31-36. IEEE, Savannah, GA, USA.

94. Pombortsis, A. and Karoulis, A., 2004. The Heuristic Evaluation of Web-Sites Concerning the Evaluators' Expertise and the Appropriate Criteria List. In *Informatics in Education – An International Journal, 3*(1), pp. 55-73. Institute of Mathematics and Informatics, Vilnius, Lituanie.

95. Potosnak, K., 1998. Conceptual Design: From User Requirements to User Interface. In *CHI 98 conference summary on Human factors in computing systems*. ACM, New York, NY, USA.

96. Redondo, M.A., Ortega, M., Molina, A.I., Bravo, C. and Sanchez-Villalon, P. HCI Curricula in the Faculty of Computer Science and Engineering at Castilla-La Mancha University (Spain): Overview and New Proposal.

97. Reed, D. and Davies, J., 2005. The Convergence of Computer Programming and Graphic Design. In *Journal of Computing Sciences in Colleges, 21*(3), pp. 179 - 187. Consortium for Computing Sciences in Colleges, USA.

98. Reimer, Y.J. and Douglas, S.A., 2003. Teaching HCI Design with the Studio Approach. In *Computer Science Education, 13*(3), pp. 191 -205.

99. Rocha, M. A. M., 2001. Adding human computer interaction studies into the informatics and computing engineering bachelor degrees in Latin America. In *CHI '01 extended abstracts on Human factors in computing systems*. ACM, New York, NY, USA.

100. Rosson, M. B., Carroll, J. M. and Rodi, C. M., 2004. Case Studies for Teaching Usability Engineering. In *Proceedings of the 35th SIGCSE technical symposium on Computer science education*. ACM, New York, NY, USA.

101. Rotard, M., Weiskopf, D. and Ertl, T., 2005. A Combined Introductory Course on Human-Computer Interaction and Computer Graphics. In *Computers and Graphics, 29*(2), pp. 267-272. Pergamon Press, Inc., Elmsford, NY, USA.

102. Rozanski, E. P. And Schaller, N. C., 2003. Integrating Usability Engineering into the Computer Science Curriculum: A Proposal. In *Proceedings of the 8th annual conference on Innovation and technology in computer science education*. ACM, New York, NY, USA.

103. Sanders, D., 2005. Incorporating Human-Computer Interaction into an Undergraduate Curriculum. In *Journal of Computing Sciences in Colleges, 20*(4), pp. 92-97. Consortium for Computing Sciences in Colleges, USA.

104. Seffah, A., 1999. Training Developers in Critical Skills. In *IEEE Software, 16*(3), pp. 66-70. IEEE Computer Society Press, Los Alamitos, CA, USA.

105. Seffah, A. and Andreevskaia, A., 2003. Empowering Software Engineers in Human-Centered Design. In *Proceedings of the 25th International Conference on Software Engineering*. IEEE Computer Society Washington, DC, USA.

106. Scholtz, J. and Laskowski, S., 1998. Developing usability tools and techniques for designing and testing web sites. In *Proceedings of the Fourth Conference on Human Factors & the Web*.

107. Sears, A. and Hess, D.J., 1999. Cognitive Walkthroughs: Understanding the Effect of Task-Description Detail on Evaluator Performance. In *International Journal of Human-Computer Interaction, 11*(3), pp. 185-200. Lawrence Erlbaum Associates, Inc.

108. Shumba, R., 2006. The Development of a Human Computer Interaction Course at a Senior Synthesis Course. In *ACM SIGCSE Bulletin archive, 38*(2), pp. 102 - 104. ACM, New York, NY, USA.

109. Skov, M. B. and Stage, J., 2004. Integrating Usability Design and Evaluation: Training Novice Evaluators in Usability Testing. In *Proceedings of the Workshop on Improving the Interplay between Usability Evaluation and User Interface Design, NordiCHI*.

110. Skov, M. B. and Stage, J., 2005. Supporting Problem Identification in Usability Evaluations. In *Proceedings of the 17th Australia conference on Computer-Human Interaction*. Computer-Human Interaction Special Interest Group (CHISIG) of Australia, Narrabundah, Australia.

111. Skov, M. B. and Stage, J., 2008. Direct Integration: Training Software Developers and Designers to Conduct Usability Evaluations. In *Proceedings of the First Workshop on the Interplay between Usability Evaluation and Software Development*. CEUR-WS.org.

112. Slavkovic, A. and Cross, K., 1999. Novice Heuristic Evaluations of a Complex Interface. In *CHI '99 extended abstracts on Human factors in computing systems*. ACM, New York, NY, USA.

113. Slone, D.J., 2009. A Methodology for Measuring Usability Evaluation Skills Using the Constructivist Theory and the Second Life Virtual World. In *Journal of Usability Studies, 4*(4), pp. 178 – 188. Usability Professionals' Association, Bloomingdale, IL, USA.

114. Stoll, P., Bass, L., Golden, E. and John, B. E., 2008. Preparing Usability Supporting Architectural Patterns for Industrial Use. In *Proceedings of the First Workshop on the Interplay between Usability Evaluation and Software Development*. CEUR-WS.org.

115. Stoll, P., Bass, L., Golden, E. and John, B. E., 2009. Supporting Usability in Product Line Architectures. In *Proceedings of the 13th International Software Product Line Conference (SPLC)*.

116. Todd, E.G., Kemp, E.A. and Phillips, C.P., 2009. Introducing Students to UI Patterns. In *Proceedings of the 10th International Conference NZ Chapter of the ACM's Special Interest Group on Human-Computer Interaction*. ACM New York, NY, USA.

117. Tscheligi, M. and Giller, V., 1995. The Gear Model of HCI Education. In *CHI '95: Conference companion on Human factors in computing systems*. ACM, New York, NY, USA.

118. Van Der Veer, G.C., 2009. Between the Ivory Tower and Babylon – Teaching Interaction Design in the 21st Century. In *P. Kotzé. W. Wong, J. Jorge, A. Dix and P. Silva (eds.), Creativity and HCI: From Experience to Design in Education*. Springer, Boston, MS, USA.
119. Van Der Veer, G.C. and Van Vliet, H., 2003. A Plea for a Poor Man's HCI Component in Software Engineering and Computer Science Curricula. In *Computer Science Education, 13*(3), pp. 207-226.
120. Webb, B.R., 1994. Teaching Usability to Visual Basic Programmers: A Case Study. In *Transactions on Information and Communications Technologies volume 7*. WIT Press.
121. Weinberg, J.B. and Stephen, M.L., 2002. Participatory Design in a Human-Computer Interaction Course: Teaching Ethnography Methods to Computer Scientists. In *Participatory Design in a Human-Computer Interaction Course: Teaching Ethnography Methods to Computer Scientists*. ACM, New York, NY, USA.
122. Wesson, J., 2006. Teaching HCI from an Iterative Design Perspective. In *The First Joint BCS/IFIPWG13.1/ICS/EU CONVIVIO HCI Educators' Workshop*. Limerick, University of Limerick.
123. Wesson, J., 2007. Teaching Creative Interface Design: Possibilities and Pitfalls. In *HCI Educators 2007, Annual International Conference of Human-Computer Interaction Educators*.
124. Wharton, C., Bradford, J. and Franzke, M., 1992. Applying Cognitive Walkthroughs to more Complex User Interfaces: Experiences, Issues, and Recommendations. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, New York, NY, USA.
125. Willshire, M.J., 2001. A Usability Focus for an HCI Project. In *Journal of Computing Sciences in Colleges, 17*(2), pp. 50 - 58. Consortium for Computing Sciences in Colleges, USA.
126. Wilson, E.V. and Germonprez, M., 2006. An HCI-Oriented Approach to the Introductory IS Programming Course. In *Proceedings of MWAIS*.
127. Wright, P.C. and Monk, A.F., 1991. The Use of Think-Aloud Evaluation Methods in Design. In *ACM SIGCHI Bulletin archive, 23*(1), pp. 55 - 57. ACM, New York, NY, USA.
128. Wright, T., Noble, J. and Marshall, S., 1997. Using a System of Tutorials and Groups to Increase Feedback and Teach User Interface Design. In *Proceedings of the 7th Australasian conference on Computing education*. Australian Computer Society, Inc, Darlinghurst, Australia.
129. Yesilada, Y., Brajnik, G. and Harper, S., 2009. How Much Does Expertise Matter?: A Barrier Walkthrough Study with Experts and Non-Experts. In *Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility*. ACM, New York, NY, USA.